SEGA® COMPUTER

The Official Sega User Club Magazine

JAN — FEB and MAR — APRIL 1986
DOUBLE ISSUE

PLENTY OF PROGRAMS!!
3D Noughts and Crosses
Basketball
Awari
Special Programming Techniques
Boxing
Sprite Generator

SPECIAL FEATURE:—
THE FORTH PROGRAMMING LANGUAGE

And Much, Much, More . . .

INTRODUCTION

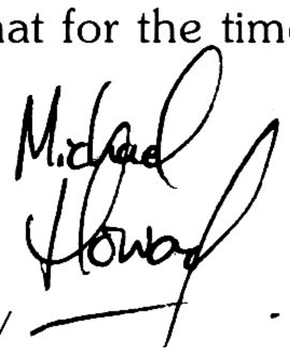
After a great power struggle that makes Dallas, Dynasty and Falcon Crest look wet, I have at last got control of this part of "The Mag."

Steve has gone back on the road as a rep for us, a job he loves, thus leaving poor old me to guard the fort!

Well, enough of the waffle and on with the show! This months isue has a special feature on adventures, for some unknown reason people of late seem to be getting into these games for a little mental stimulation, and I'm glad to say that many software houses are writing them to keep up with the demand.

Also in this issue is a special look at random numbers, this topic is a little more technical, but not TOO technical, so hopefully those of you who are not of a technical nature might find it a little edible, and those who are quite technical should find it quite enjoyable.

Anyway, that's that for the time being, I hope you enjoy The Mag.



Michael Howard
PROGRAMMER/
TECHNICAL SUPPORT

IMPORTANT NOTICE

Please note that this magazine is actually two magazines in one. We apologise for it's late arrival, but it was beyond our control.

It is important to note that this is **YOUR** magazine. So please send in any programs that you have, be them small or large, complex or simple ... it matters not. If someone sends in a program and someone else learns from it then it has been worth it! To be quite frank I could name ten people in the UK, and three in New Zealand who now make a lot of money through writing programs, and they all started by writing a few simple programs and having them published in computer magazines!

SO GET WRITING!

Contents

Letters to the Editor....2
Software Reviews....3
Saturn Earth Demo....4
Cryptography....5
Solitaire....6
Hurkle....8
Random Words 1/2....9
Random Numbers....10
Resistors....12
Sine Waves....13
Important Notice....14
King of the Islands....15
Star Wars....19
Russian Roulette....22
Awari....23
More Maths Fun....24
Sprite Generator....25
Beginners Box of Bugs....28
Competition....The Solution....30
Basketball....31
Buzzwords....34
3D Noughts and Crosses....35
Forth....39
Bias Dice....48
Limitless Mazes....51
Mystery....52
Adventure Games....53
Acey Ducey....54
More Maths Fun....55
Boxing....56
Bowls....58
The Bouncing Ball....60

ESTER 318

OVERSEAS SEGA USER's CLUBS

Camden Sega User's Club
Contact Steve MacDonald
2 Coolalie Ave
Camden 2570
Australia

LOCAL SEGA USERS CONTACTS

AUCKLAND CENTRAL SEGA USERS CLUB
C/o 287 Broadway Furniture
Newmarket
Contact: George Shaw
Ph. 547-543

B.O.P SEGA USERS CLUB
Sega 102B Hinewa Road
TAURANGA
Sec P.J. McDonald
Ph 64826 a/h
Tauranga

CHRISTCHURCH USER'S CLUB
Contact Graham Rudman
29 Primrose Street
CHRISTCHURCH 5

GISBORNE AREA USER'S CLUB
Trevor Gardiner
Ph. 83-068 HM
or 87-175 WK

HAMILTON SEGA USER'S CLUB
P.O. Box 1548 Hamilton
President: Mr Colin Bell 73-826
Secretary: Mrs Anne Thrush Ph 437-312
Meetings held fortnightly on Monday at Waikato Technical Institute in Room F129.

NAPIER SEGA USERS CLUB
Sec E.P. Lins
41 Higgins Street
NAPIER

PUKEKOHE SEGA USERS CLUB
C/o 4 Roose Avenue
Pukekohe
Contact: Selwyn
Ph. Pukekohe 86-583

ROTORUA SEGA USERS CLUB
C/- 61 Devon Street
ROTORUA

TOKOROA SEGA USERS GROUP
C/o 1 Pio Pio Place
TOKOROA
Contact Geoff Phone Number 67-105
Tokoroa

SOUTH CANTERBURY USERS CLUB
P.O. Box 73 Timaru New Zealand
President: Geoff McCayghan
Secretary: Lloyd Van Der Krogt
Contact Phs: 60-756, 61-412 TIMARU

SOUTH COROMANDEL USERS CLUB
P.O. Box 183
WHANGAMATA
Contact: Sid
Ph. 58-775 Whangamata

SOUTH TARANAKI MICROCOMPUTER SOCIETY
D.M. Beale
7A Clive Street
HAWERA

WELLINGTON USERS CLUB
Contact: Shaun Parsons
P.O. Box 1871
Postal Central
Wellington
Ph 897-095 a/h
Ph 727-666 work

The above are contact names and addresses for Sega Users Clubs. If you wish to have your club advertised write to Sega Users Club P.O. Box 2353, Auckland.

If you have set up a local area user's club and you would like us to publish the details concerning your club please send them to us and we will publish the information for no charge.

LETTER'S TO THE EDITOR

DEAR EDITOR

I am having a spot of bother with the program "Jet Ranger". Firstly I keep getting a Statement parameter error, and when I look at lines 1 and 2, they are all jumbled up. Could you please help.

Sid Bruce,
Wellington

DEAR SID

The error you are getting is due to a typing error on your behalf, check that all the lines with DATA at the start contain only the digits 0,1,2,3,4,5,6,7,8,9,a,b,c,d,e and f. It is possible that you have accidently placed a O (the letter) instead of 0, or perhaps an I instead of a 1.

The lines 1 and 2 should get jumbled up, as this is where the machine code for the program is stored.

DEAR EDITOR

A couple of months ago I sent in a cassette to you, and I have not received it back . . . where is it?

Henry Hunter
Dunedin

DEAR HENRY

You are not the only person suffering from this, I am sorry to say. But the reason for so many unreturned tapes is simple, I get about 20 tapes a week for review, and 19 out of that 20 are not marked, so it makes it next to impossible to identify a cassette. The moral to this story, is if you wish to send a tape for evaluation, and wish to get it back, then please put your name and address on the cassette, as well as the title of the program.

DEAR MICHAEL

I wish to set up my own software house selling some of the programs I have written, could you please tell me of how I would do this.

Jane Harrison
Browns Bay

DEAR JANE

Here is how to do it:-

1) Make sure that the program is of good quality.
2) Buy some good quality C-10 cassettes, or a size to suit. Make sure that they are of the "Low Drop Out" kind.
3) Make copies of say, 40, of the program and send one to me along with your address and I'll review it in the mag.
4) Get some inlays printed.
5) Go to your local dealers and ask them to stock the program for you, on a "Sell or return" basis. This means that if the dealer doesn't sell the program then he has the right to return the program back to you, for a refund.
6) You could perhaps get the retailer to stock the program on your behalf, in other words, he doesn't buy the program off you, he just displays it for you and he gets a small percentage at each sale.
7) Advertise in the mag. Really cheap!!

One important thing is to make sure that the program is 100% guaranteed to load!

I hope that this has given some helpful advice to budding programmers.

DEAR EDITOR

This is just a note to say that the mag has now become the way I, and many others I know, like it . . . full of programs and interesting news. Keep up the good work.

Geoff Steele
Hamilton

DEAR GEOFF

Thanks a lot for the compliments. You'll love this issue!! Have any others got any criticisms of the mag, don't forget it is your mag!

Software Reviews and News

It's funny how all of a sudden some **REALLY** great software is available, just when people were crying out for more! So without further ado, here are a couple of reviews.

AEROBAT

. . . 32k only

Just about everybody has been yelling out for a flight simulator, well here it is! This program allows you to simulate the flight of a small light aircraft called a Cessna 152 Aerobat.

When the program is running, you are confronted with a cockpit view of the aircraft this includes the view outside the cockpit, a compass, a vertical speed indicator, fuel guage, power guage, artificial horizon, an air speed indicator and an alimeter. At first you are on the ground on a runway, you slowly accelerate and the runway starts to move as you get faster and faster, then when you reach a safe enough speed, you pull back on the joystick and up you go, flying!

When in the air, you have many controls over your craft, you control the ailerons, these control the roll of the aircraft. Elevators control pitch, ie up and down movement, and the rudder controls yaw (no not yawn!) this is sideways swing. The breaks are really neat, you can sit on the runway and build up to full power, then let the breaks off and just about catapult yourself along the runway!

The most impressive part of the program has to be the graphics, everything is in full 3D wire (called Vector graphics) graphics. To the South East of the airfield is a lake and on this lake is an arched bridge, and it is in fact possible to fly under the bridge! In full 3D!

It is also possible to do aerobatics such as loops, stalls, stall turns, rolls, barrell rolls and combinations of the above.

All details are accurate such as the inability to stay upside down for too long as the fuel system does not have the ability to function upside down for prolonged lengths of time.

Note that the program is for the 32k model only.

To sum up, this is a truly remarkable game and well worth the money.

It is available now from Poseidon Software, CPO Box 784, Hamilton.

Burglar Bill

An absolutely stunning program with amazingly smooth animated graphics. If any of you have seen the program Jet Set Willy on other computers and liked it then this is the game for you. It is, of course 100% Machine Code, as are all programs these day! And this leads to a really pleasurable and enjoyable game.

The game goes something like this . . . you are a burglar called Bill, pretty obvious really! And you decide to case the place. Well that sounds quite easy, but all is not as it seems, inside the houe there are many wierd and wonderful creatures running around the place, you must dodge these and get hold of the artifacts that the house holds. This is in essence a Ladders and levels type game where you have to run upstairs and downstairs dodging this-that-and-the-other . . . in general good wholesome stuff! I can't recommend this game enough. It is available from Poseidon Software.

Sega Mon

A must for all serious programmers. A monitor is used to de-bug machine code programs and let's you patch up the mistakes. You can set registers, inspect them and test them at any time that you want. Once again this is another excellent product from Poseidon Software, although it's use will be only by proficient machine code programmers.

Sega Word 3

This is the program that all you disc drive owners have been waiting for . . . a professional word processor. This program let's you copy paragraphs, delete paragraphs, format the document, force end of pages, display menus, set printer controls, kill documents, set margins, move paragraphs, print documents, read in documents

and save documents, search for a word etc et. The disc that comes with the program has a bonus program on it for Cashbooks, it is only the instructions for using the program and the cashbook is not in the country yet. Getting back to the Word processor, it is a breeze to use and extremely quick. In fact I would say that this is one of the best wordies this side of Pluto!

For more info get in contact with Poseidon Software (again!!)

Saturn Earth Demo

by W.G. Wells

You may remember that in the last magazine there was a little program that let you draw a small picture of the planet Saturn. Well, this program is a modification of that.

Mr Wells reckons that he has learned a heck of a lot from re-writing the program, which as far as I'm concerned is what this mag is all about!

```
10 SCREEN 2,2:CLS
20 VS=1
30 COLOR 1,1,(0,0)-(255,191),1
40 FOR E=1 TO 200:PSET(RND(1)*255,RND(1)*191),15:NEXT E
50 CIRCLE (31,30),10,5,1,0,1,BF
60 CIRCLE (31,30),20,15,.2,.85,.65
70 COLOR 5:CURSOR 100,180:PRINT CHR$(17);" SEGA ":CURSOR 164,180:COLOR3:PRINT CH
R$(16);"EARTH DEMO"
71 CIRCLE (230,160),10,11,1,0,1,BF
72 CIRCLE (230,160),20,15,.2,.85,.65
73 CIRCLE (230,30),10,13,1,0,1,BF
74 CIRLCE (230,30),20,15,.2,.85,.65
75 COLOR 9:CURSOR 100,25:PRINT CHR$(17);"SATURN"
76 CIRCLE (30,160),10,9,1,0,1,BF
77 CIRCLE (30,160),20,15,.2,.85,.65
80 COLOR 11,1,,1
90 X=128:Y=90:IN=.08:R2=28
100 FOR R1=100 TO 65 STEP -5
110 R2=R2-1
120 FOR TH=0 TO 2*PI-IN STEP
IN
130 LINE(R1*COS(TH)+X,R2*SIN(TH)+Y)-(R1*COS(TH+INC)+X,R2*SIN(TH+INC)+Y)
140 NEXT TH
150 LINE-(R1+X,Y)
160 NEXT R1
170 BCIRCLE(128,90),50,,1,.5,1,BF
180 R2=50:ST=.015:CIRCLE(128,90),50,,1,1,0
190 FOR R1=46 TO 16 STEP -4
200 FOR TH=PI/2 TO 1.5*PI STEP ST
210 PSET(R1*COS(TH)+X,R2*SIN(TH)+Y)
220 NEXT TH:ST=ST+.009:NEXT R1
230 GOTO 230
```

..oOo..

Secret Writing . . . Explore the craft of cryptography

Michael Howard

Secret writing, or cryptography, is the craft of hiding information by by converting into info into a code that looks really ridiculous, thus leading the person who is being exceptionally nosey to think that the text is a load of old cobblers. Only by using the correct password can the reader unlock all the secrets that you placed in the message.

This month I am going to present a program that turns your Sega into a coding machine, using a simple, but effective, and acient cryptographic technique known as Caesar cipher.

The Caesar cipher got it's name from the ol' Roman called Julius Caesar (you know "Ides of March" and all that!). The way the cipher works is quite simple, basically, every letter is shifted by a certain amount, the amount is called the key. For instance, with a key of say, 3, the letter A is shifted forward to become D, and B becomes E etc. If we reach the end of the alphabet, then we simply continue to the start of the alphabet. Thus, using the same key of 3, the letter Y becomes B. Keeping with a key of 3, the message:

THE APPLES ARE RIPE

becomes the coded message:

WKH DSSOHV DUH ULSH

To decode the message, we shift each and every letter in the coded message back the same key. In this case, we shift W back 3 to get T, K back 3 to get H and so on and so forth.

Unfortunately, the shifting of letters back and forward is extremely boring and tiresome. But lo and behold, computers are designed to do boring and repetitive tasks. So here follows a short prelude of how the program works.

Let's start with a short description of the cipher machines duties. First it asks you to type in the key, select encoding or decoding, and type in the message. The Sega picks out the first letter of the message, encodes or decodes that letter, and prints the coded letter. The computer picks out the next letter in the message (ignoring all spaces and punctuation) and encodes or decodes that letter. It repeats the process until no more letters are left to be coded.

The program

The program is quite straight forward and is written entirely in BASIC. It uses four special BASIC functions to accomplish the ciphering. All of these functions perform operations on strings (a sequence of letters and other characters).

LEN measures the number of characters in a string. For instance, PRINT LEN(M$) prints the length of the string M$. The program uses LEN to determine the number of letters in the message to be coded.

**MID$** picks out one or more letters from a string. For instance, MID$ (message, position, length) gives us a portion of the message starting at the stated position and running for the stated length. Eg;

If M$="HELLO MUM!"

MID$(M$,2,9) = "ELLO MUM!"
MID$(M$,3,3) = "LLO"

In the program, MID$ picks out one at a time the letters of the message to be coded.

ASC gives us the numeric position of the character in the Sega's alphabet. For instance ASC("A") gives 65, this is the position of the letter "A". This may seem illogical, but don't forget, your micro computer has a vast alphabet of 256 letters and characters, the letter "A" is number 65 in that alphabet. The alphabet that is in the computers memory is called ASCII (American Standard Code for Information Interchange. Other examples are ASC("M") is 77.

**CHR$** is the opposite function of ASC, you supply a number and it gives the character. EG CHR$(65) gives the letter "A".

The following variables have been used in the program:

VARIABLE	DEFINITION
LB	ASCII position of letter "A"
UB	ASCII position of letter "Z"
K	key
CH$	choice of encoding or decoding
M$	message
P	indicates current letter
X$	letter pointed to by P
Y$	encoded or decode letter
YP	ASCII position of Y$

The program is listed below:

```
1 REM MH
10 lb=65
20 ub=90
30 PRINT "Secret Writing Machine"
40 PRINT
50 INPUT "Enter the key (1-25) ";k
60 IF k<1 OR k>25 THEN 50
70 INPUT "Enter E to encode, or D to decode ";ch$
80 IF ch$<>"e" AND ch$<>"E"AND ch$<>"d"AND ch$<>"D" THEN 70
90 PRINT"Enter a message"
100 INPUT m$
110 IF m$="" THEN 90
```

```
120 FOR p=1 TO LEN(m$)
130 x$=MID$(m$,p,1)
140 IF x$>"A" AND x$<"Z" THEN 170
150 y$=x$
160 GOTO 240
170 IF ch$="D" OR ch$="d" THEN 210
180 yp=ASC(x$)+k
190 IF yp>ub THEN yp=yp-26
200 GOTO 230
210 yp=ASC(x$)-k
220 IF yp<lb THEN yp=yp+26
230 y$=CHR$(yp)
240 PRINT y$;
250 NEXT p
260 PRINT
270 PRINT"Le End (more French!)"
280 END
```

Solitaire

In this game or puzzle, call it what you like, 48 checkers are placed on the two outside spaces of a standard 64 square chess board as is shown in the program.

The object of the game is to remove as many checkers as possible by diagonal jumps (as in Draughts or Checkers).

It is easy to remove 30 to 39 checkers, a challenge to remove 40 to 44 and you must be brilliant to remove 45 to 47.

The numbering of the board is as follows:-

1	2	3	4	5	6	7	8
9	10	11	12	13	14	15	16
17	18	19	20	21	22	23	24
25	26	27	28	29	30	31	32
33	34	35	36	37	38	39	40
41	42	43	44	45	46	47	48
49	50	51	52	53	54	55	56
57	58	59	60	61	62	63	64

And here are the opening positions:

1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	0	0	0	0	1	1
1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1

Note that a 1 denotes a checker a 0 shows a space.

```
10 REM MH
20 SCREEN 1,1:CLS
30 PRINT"Solitaire Checkers"
40 DIM a(64)
50 PRINT"Here is the board.":PRINT
60 FOR j=1 TO 57 STEP 8
70 FOR i=j TO j+7
80 x=i-j+1:PRINT TAB(x*4);i;
90 NEXT:PRINT
100 NEXT
110 PRINT:PRINT"And here is the opening position."
120 PRINT
130 FOR j=1 TO 64
```

```
140 a(j)=1
150 NEXT j
160 FOR j=19 TO 43 STEP 8
170 FOR i=j TO j+3
180 a(i)=0
190 NEXT i,j
200 m=0
210 GOTO 460
220 INPUT "Jump from";f
230 IF f=0 THEN 520
240 INPUT "to";t
250 PRINT
260 f1=INT((f-1)/8)
270 f2=f-8*f1
280 t1=INT((t-1)/8)
290 t2=t-8*t1
300 IF f1>7 THEN 400
310 IF t1>7 THEN 400
320 IF f2>7 THEN 400
330 IF t2>7 THEN 400
340 IF ABS(f1-t1)<>2 THEN 400
350 IF ABS(f2-t2)<>2 THEN 400
360 IF a((t+f)/2)=0 THEN 400
370 IF a(f)=0 THEN 400
380 IF a(t)=1 THEN 400
390 GOTO 420
400 PRINT"Illegal move. Try again.."
410 GOTO 220
420 a(t)=1
430 a(f)=0
440 a((t+f)/2)=0
450 m=m+1
460 FOR j=1 TO 57 STEP 8
470 FOR i=j TO j+7
480 x=i-j+1:PRINT TAB(x*4);a(i);
490 NEXT:PRINT
500 NEXT
510 PRINT:PRINT:GOTO 220
520 s=0
530 FOR i=1 TO 64
540 s=s+a(i)
550 NEXT i
560 PRINT"You made";m;" jumps":PRINT"and had";s;" pieces left on the board"
570 PRINT
580 PRINT"Nuvver go?"
590 a$=INKEY$:IF a$="" THEN 590
600 IF a$="y" OR a$="Y" THEN 20
610 PRINT
620 PRINT"Okay, I hpoe you enjoyed it!"
```

Hurkle

A hurkle is a happy beast and lives in another galaxy on a planet named Esylu that has three moons. Hurkle are favourite pets of the Anj, the dominant race of Esylu, okay enough of the waffle!

In this program a shy hurkle is hiding on a 10 by 10 grid. Homebase is a point in the bottom left hand corner, and it has the coordinates 0,0. It is up to you to try and find the beast on this grid. Your guess as to where the beast lies should be a pair of whole numbers, separated by a comma. After each try, the computer will tell you if you are too far North/South/East or West. You have 5 guesses to find him; you may change this number by altering line 40 accordingly.

```
10 CLS
20 PRINT"The Hurkle.."
30 PRINT:PRINT:PRINT
40 n=5
50 g=10
60 PRINT
70 PRINT"The Hurkle is hiding in a";g;"by";g
80 PRINT
90 a=INT(g*RND(8))
100 b=INT(g*RND(8))
110 FOR k=1 TO n
120 PRINT"Guess #";k;
130 INPUT x,y
140 IF ABS(x-a)+ABS(y-b)=0 THEN 230
150 REM
160 GOSUB 250
170 PRINT
180 NEXT k
190 PRINT:PRINT"Sorry, that's";n;" guesses."
200 PRINT"The Hurkle is as";a;",";b
210 PRINT
220 GOTO 90
230 PRINT"You've found him in";k;" guesses!"
240 GOTO 210
250 PRINT"Go ";
260 IF y=b THEN 310
270 IF y<b THEN 300
280 PRINT"South ";
290 GOTO 310
300 PRINT"North ";
310 IF x=a THEN 360
320 IF x<a THEN 350
330 PRINT"West ";
340 GOTO 360
350 PRINT"East ";
360 PRINT
370 RETURN
```

Random Words Program 1

Have you ever written a program, and then thought, "Now what could I call the main character?", or "What should I call the land", I'm sure you know what I mean. Well, hopefully, this little gem of code will help you in deciding such nomeclature (a fancy name for naming!).

All you do is enter the program and run it, after a word is printed on the screen just press a key and another word will be printed. The program as it stands will probably not give many words, but I guarantee that it will give you, some good ideas for new words.

```
10 x=RND(-1)
20 DEF FN rn(x)=INT(RND(8)*x)+1
30 DATA b,c,d,f,g,h,j,k,l,m,n,p,q,r,s,t,v,x,w,y,z,ch,sh,st,ts,ny,yn,sl,nd,-1
40 DATA a,e,i,o,u,au,ea,ae,ee,aa,ou,io,oi,oo,-1
50 DIM a$(30),b$(30)
60 RESTORE
70 i=1
80 READ d$:IF d$="-1" THEN 100
90 a$(i)=d$:i=i+1:GOTO 80
100 j=1
110 READ d$:IF d$="-1" THEN 130
120 b$(j)=d$:j=j+1:GOTO 110
130 d$="":s=FN rn(2)
140 FOR le=1 TO FN rn(3)+1
150 IF s=1 THEN d$=d$+a$(FN rn(i))+b$(FN rn(j)):GOTO 170
160 d$=d$+b$(FN rn(j))+a$(FN rn(i))
170 NEXT
180 PRINT d$
190 IF INKEY$="" THEN 190
200 GOTO 130
```

Random Words Program 2

Here is another program for generating random words. Just run the program, and it will produce a matrix of letters, now just link together some groups. of letters to make up a word.

In my opinion, this program is better than the first, due to it's lack of complexity, flexibility and ease of use.

```
10 CLS
20 FOR a=0 TO 20
30 FOR b=0 TO 30
40 PRINT CHR$((RND(8)*25)+65);
50 NEXT:PRINT:NEXT
```

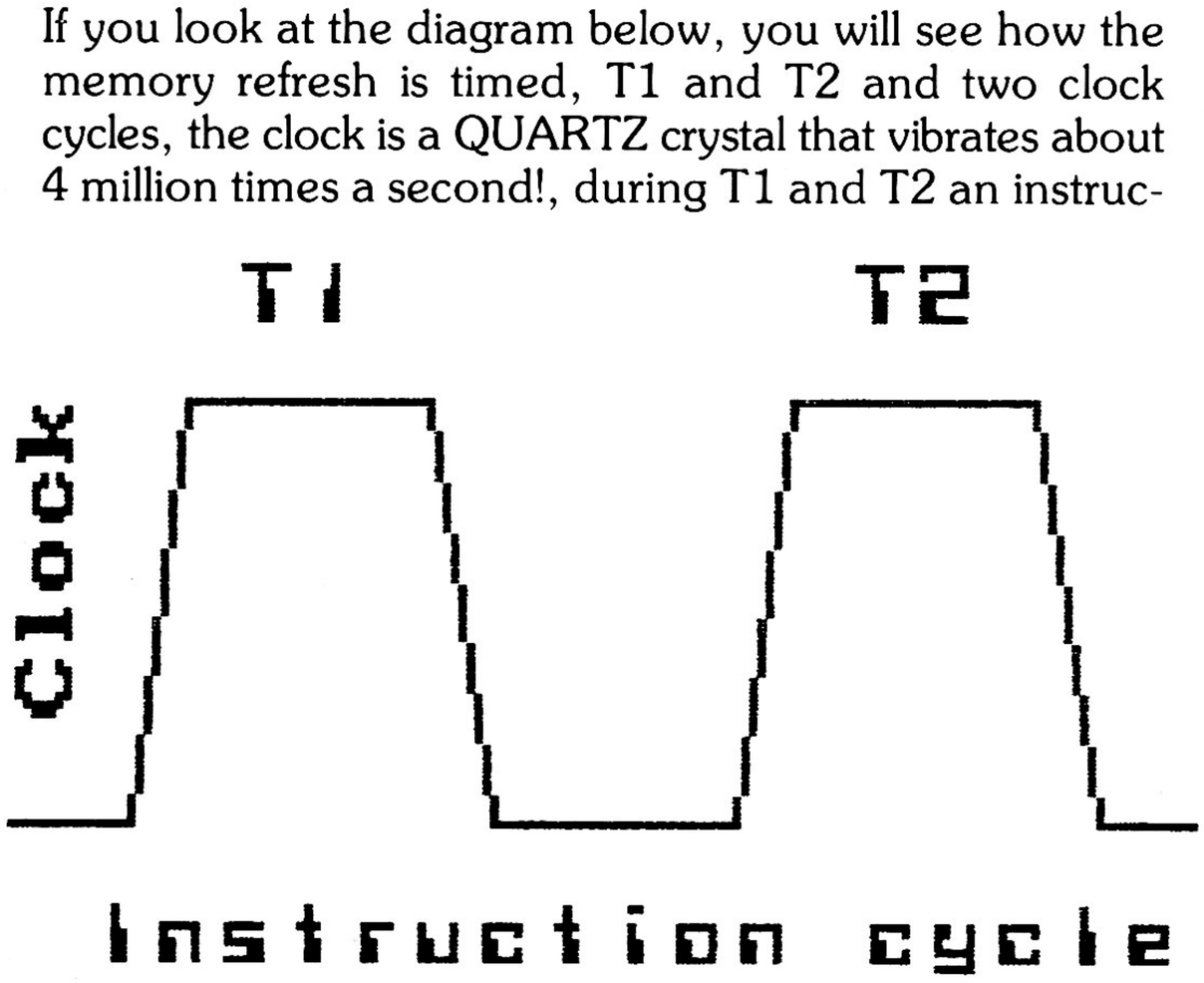
Random Numbers from Machine Code

. . . Michael Howard

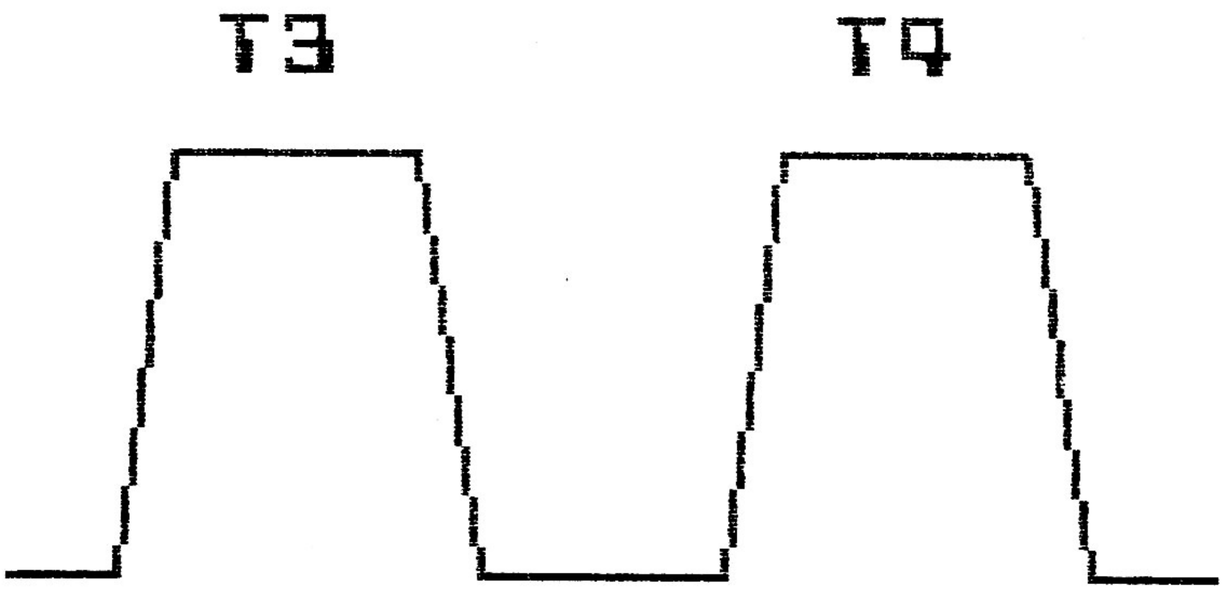
Generating random numbers from machine code is not at all easy, and Z80 users may be tempted to seek a solution by using the memory refresh register to generate them. But firstly, let me explain a little about memory and how it works (I bet you're wondering what memory has to do with random numbers!).

Compute memory is divided into two parts, ROM and RAM. The Read Only Memory is the Sega's "handbook", every program that you write has to be cross referenced with the ROM to see what to do next and how to execute every BASIC instruction that you write. The RAM or Random Access Memory, is where your programs are stored. Now RAM can be broken down into two types, Static and Dynamic. Whenever you store some data or a program in static RAM, the data stays there until you turn the power off, the disadvantage of static RAM is that the chips are quite large, so they are not used all that often. On the other hand, dynamic RAM chips are small, and can hold a lot of information on each chip. The only drawback of dynamic RAM chips is that they "lose" their memory after about 1 second! So every 50th of a second, the Z80 inside your Sega "updates" the contents of some of the RAM (because all the RAM in your computer is dynamic), this "updating" is called REFRESHING. Now the Z80 has to keep track of what part of the memory it is refreshing so that it does not refresh one part of memory too often and another part never gets refreshed. To keep track of where the Z80 is refreshing, the "memory refresh register" is used. The Refresh Register is in fact an 8-bit register just like registers "A","B","C" etc etc . . . but only 7 bits are used, after every machine code instruction is executed the contents of the Refresh register are incremented by one, then that bank of memory is refreshed. As a very simple example, if R (the refresh register) holds a value of 42, then bank 42 will be refreshed. When the next instruction is executed, R will hold 43 and bank 43 will be updated, and so on and so on.

If you look at the diagram below, you will see how the memory refresh is timed, T1 and T2 and two clock cycles, the clock is a QUARTZ crystal that vibrates about 4 million times a second!, during T1 and T2 an instruction is operated on by the Z80 (the CPU inside your Sega, CPU means Central Processing Unit), but during T3 and T4, dynamic memory chips are refreshed.



Instruction cycle



Memory refresh cycle

Now, as I stated before, you are probably wondering what on Earth, memory has got to do with random numbers, well the answer is the R register. The intended use of the register is to keep track of the memory refreshing, but it is possible to be read the register by use of the LD A,R instruction. The result in bits 0-6, will be a number in the range 0-127. If the read is called at a moment dependent on some action outside the processinf system, such as a key being pressed, the number obtained will vary at random.

For most purposes, however, a 0-127 range is excessive. If we want a number in the range 1-6, to simulate a dice being thrown, you could use AND 7 to cut the range down to 0-7 and ignore values of 0 and 7 and looping back to repeat the LD A,R instruction.

You could do that, but it would be unwise. Think about it. The loop to the repeat of LD A,R will contain a predictable number of instructions, so the change in the R register will be equally predictable, and the last thing we want with random numbers is predictibility! What we could do is as I mentioned before, is read the keyboard for any action and when a key is pressed get a number from the R register, but this is really clumsy, without an outside trigger such as a key being pressed, no numbers will be generated. So what do we do?

BREAK THE LOOP

All this messy random number business leads to only one efficient solution. If we want a number in the range 1-6, we would multiply the memory refresh register with 6, divide by 128 and add 1 to the result, so a bit complex? Not really! The little program below shows how it is done in machine code:

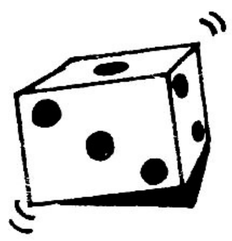
```
                nmemonics               hex data
                LD HL,0000              21,00,00
                LD D,0                  16,0
                LD E,N                  1E,N
                LD A,R                  ED,5F
                ADD A,A                 87
                LD B,7          06,7
:--->           RLC A                   07
:               JR NC           >---:   30,1
:               ADD HL,DE         :     19
:               ADD HL,HL       <----   29
---<            DJNZ                    10,F9
                LD A,H                  7C
                INC A                   3C
                RET                     C9
```

In this program, the number range is placed in the E register. So if we want numbers in the range 1-6, then we would load E with 6, ie LD E,6 and a number in the range 1-6 will be in the A register (accumulator).

So all in all this is a really neat little random number generator that is extremely versatile and efficient.

Well that just about covers the random number subject in machine code.

SIMULATING DICE THROWS



A final point on random numbers concerns the simulation of dice throws. If two dice are to be thrown, it is not correct to generate a number from 1 to 12, with a equal value of each. Two random numbers, each in the range 1-6, must be generated, and their sum is the new random number. Why? As Fig 2 shows, there is only one chance in 36 of getting a 2 or 12 whilst there are six chances of getting a 7.

Random numbers are useful, sometimes even essential, but you have to think about them very carefully at times so make sure that you know what you are doing when messing about with them.

1st dice

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	10
5	6	7	8	9	10	11
6	7	8	9	10	11	12

Fig 2. Two dice combinations

PS: I know that this article may seem a little heavy, but it has to be really, so please accept my apologies all those who know not of Machine Code!

COUNT THOSE BANDS ON RESISTORS

Many people have a lot of trouble trying to work out what those bands mean on resistors, sometimes they can be confusing and lead to lots of human error. So I thought I would write a program to allow you to convert the coloured bands on resistors to a more digestible form.

When you run the program you will be asked to input the three band colours, when you have entered the relevant data the computer will give you the resistance of that resistor and it's tolerance.

If you want to remember the order of the colours remember this little rhyme:

"Black Baron Runs Over Young Girl Bellowing Vicious and Gruesome Words"

All you electricians out there should find this routine extremely useful.

Just for general interest here is a little note for those who don't know about resistors.

A resistor is a small cylindrical shaped device that has two wires popping out of the end. They are used in electronic circuits to stem the flow of an electric current, the reasons for doing this are quite complex at times. Now, sometimes you may wish to stem the flow to a high degree, and sometimes to a small degree, and to alter the flow, you just alter the resistance of the resistor, the higher the resistance, the less power gets through, and vice versa. The way you can tell the resistance is by the banding on the resistor, certain combinations of colours give certain resistance values. Now, one of the problems is reading those bands of colour and remembering the values, so that is what this program is all about.

For referance, bands 1 and 2 determine the resistance, and band 3 the tolerance ie, how accurate the value given on the resistor is.

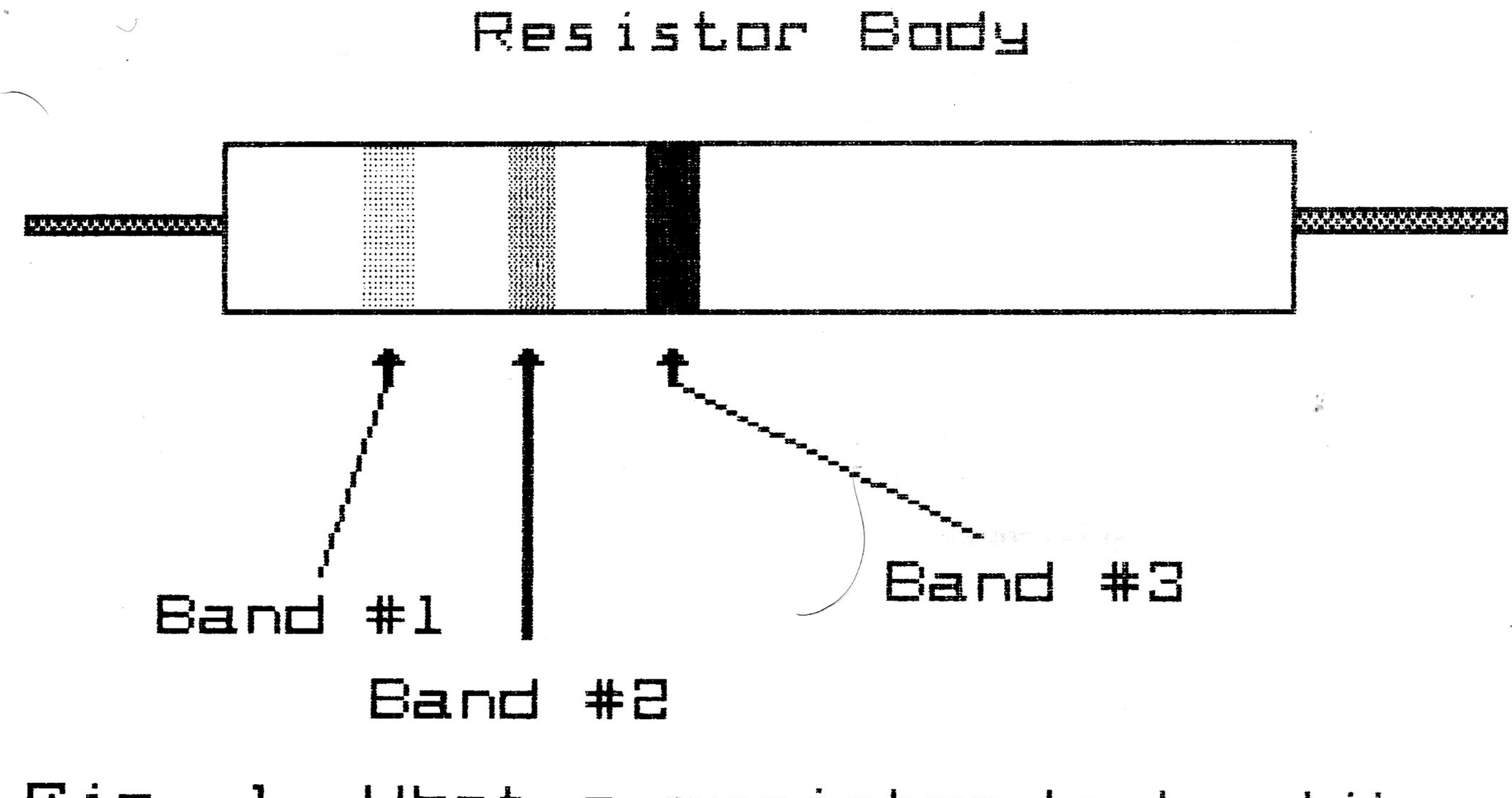


Fig. 1 What a resistor looks like.

The colour of the Bands determine the resistance of the resistor.

```
RESISTOR

10 DATAblack,brown,red,orange,yellow,gre
en,blue,violet,grey,white
20 DIMA$(9),B$(9),C$(9)
30 FORI=0TO9:READA$(I):B$(I)=A$(I):C$(I)
=A$(I):NEXTI
40 CLS:PRINTCHR$(20)
50 PRINT"Colour Resistor Codes":PRINT
60 INPUT"What is the 1st colour ";A$
70 INPUT"2nd colour ";B$
80 INPUT"3rd colour ";C$
90 GOSUB190
100 D=((A*10)+B)*C
110 IFD<1000THEN140
120 IFD<1000000THEN150
130 PRINT:PRINT"Value";D/1ǫ000000;" M.Ohm
s":GOTO160
140 PRINT:PRINT"Value";D;" Ohms":GOTO160
150 PRINT:PRINT"Value";D/1000;" k.Ohms"
160 PRINT"Press a key"
170 IFINKEY$=""THEN170
180 GOTO40
190 FORI=1TO9
200 IFA$(I)=A$ THENA=I
210 IFB$(I)=B$ THENB=I
220 IFC$(I)=C$ THENC=10^I
230 NEXT
240 IFA$="black"THENA=0
250 IFB$="black"THENB=0
260 IFC$="black"THENC=1
270 IFC$="gold"THENC=.1
280 IFC$="silver"THENC=.01
290 RETURN
```

The Sine Wave

Did you ever go to a computer show and see a computer waiting to do a demo for you? It was one of these moments that made me decide to do my own simple, yet interesting demo. Here it is:

If you have the SP-400 printer plotter then alter line 130 to:-

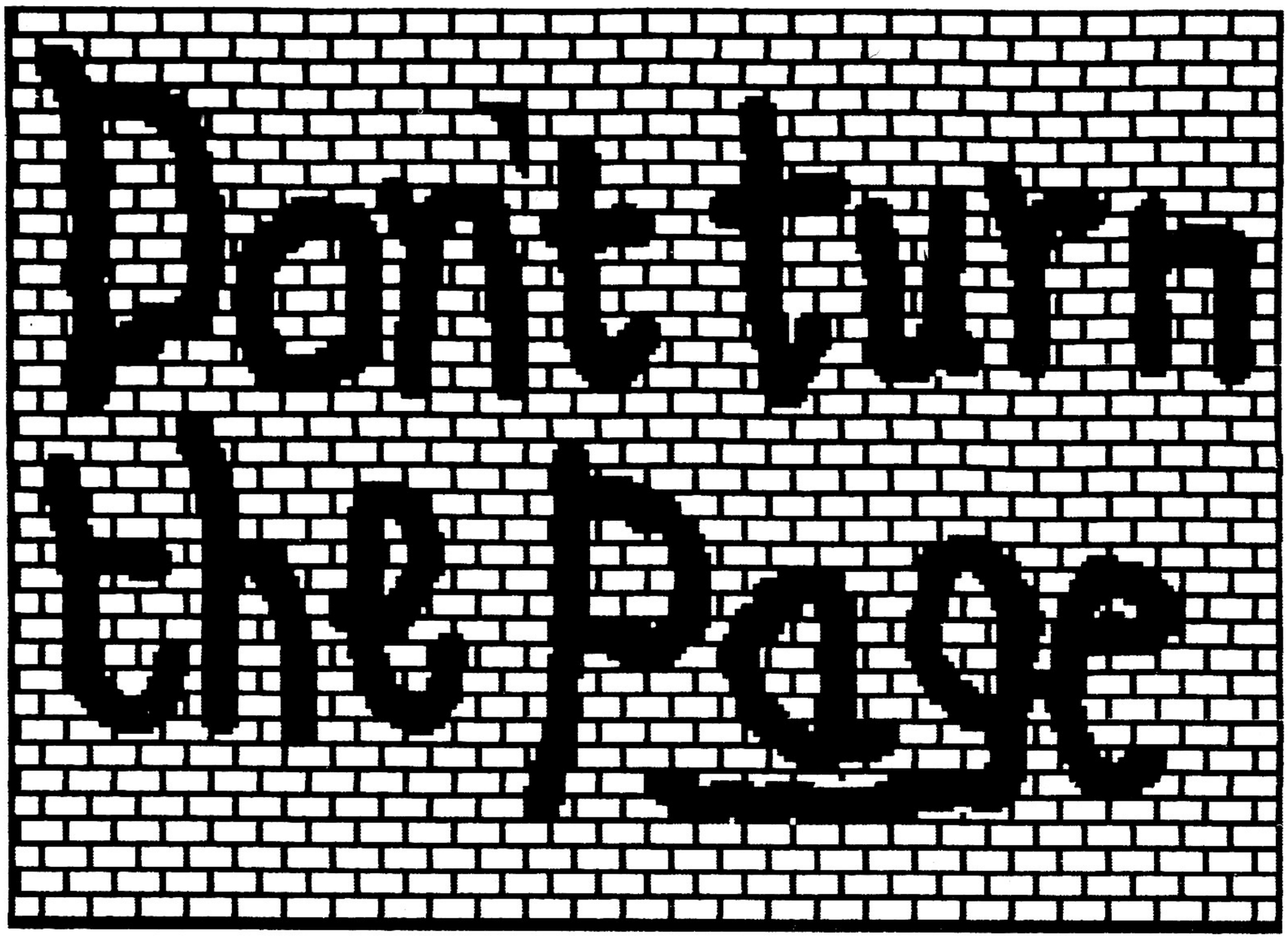
130 LPRINT TAB (A);

Try to limit your texts to less than 6 characters.

```
10 INPUT "ENTER A WORD.........";A$
20 INPUT "NOW ENTER ANOTHER...";B$
30 B=0
40 FOR T=0 TO 40 STEP 0.25
50 A=INT(15+15*SIN(T))
60 PRINT TAB(A);
70 IF B=1 THEN 110
80 PRINT A$
90 B=1
100 GOTO 130
110 PRINT B$
120 B=0
130 NEXT T
```

If you have the SP-400 printer plotter then alter line 130 to:-

130 LPRINT TAB(A);



an important notice

As you may have noticed, this issue is a bit late. Well the reason for this is simple.....

There is a lack of programs from you lot!!

So PLEASE, please, dlaesa, please send some programs in! The more you send in the better.. It's your way to fame!!

You can turn the page NOW!!!!

King of the Islands

Phil Tottle

This is one of the more comprehensive, difficult and interesting land resource management games going! If you have never played this type of game before, then I suggest that you start off on an easier one, like the game in **More than 50 programs for the Sega SC-3000.**

In this game, you are the King (or Queen for that matter!) of a small island called **Esterlouisia,** which measure 70 by 30 kilometres. Your job as the boss is to decide upon the budget of the country and distribute money to your countrymen from the communal treasury.

The money system is Vanders; and each person needs at least 100 vanders a year to survive. Your country's income comes from farm produce and tourists visiting your superb forests, fishing and hunting facilities. Part of your land is arable farm land but it also has an excellent mineral and may be sold to foreign industry for strip mining. Industry import and support their own workers. Crops cost between 10 and 15 vanders per square mile to plant, cultivate and harvest. Your goal is to complete an 8-year term of office without major mishap. Good luck. A word of warning though . . . it isn't easy!!!!!!!

```
10 REM by Phil Tottle
20 CLS:a=RND(-1)
30 PRINT"The King of the Islands"
40 PRINT"Do you want instructions?"
50 a$=INKEY$:IF a$="" THEN 50
60 n5=8
70 IF a$="y" OR a$="Y" THEN 90
80 GOTO 270
90 PRINT:PRINT:PRINT
100 PRINT"Congratulations you've just become"
110 PRINT"The boss of the small communist island"
120 PRINT"called 'EsterLouisia'."
130 PRINT"Your job is to decide on the economy of"
140 PRINT"the country as well as the budget."
150 PRINT" You have to distribute money from the"
160 PRINT"country's treasury. The money system is called"
170 PRINT"'VanDers', and each person needs 100"
180 PRINT"100 VanDers a year to survive."
190 PRINT"Your country's income comes from"
200 PRINT"farms and tourism."
210 PRINT"Half of your land is farm land which"
220 PRINT"also has an excellent mineral content"
230 PRINT"and may be sold to foreign industry"
240 PRINT"for strip mining. Crops cost 10 to"
250 PRINT"15 VanDers per sq mile to plant."
260 PRINT"Your goal is to stay in office for"
270 PRINT n5;" years"
280 PRINT:PRINT:PRINT"Good Luck! You'll need it!"
290 IF INKEY$="" THEN 290
300 a=INT(60000+(1000*RND(8))-(1000*RND(8)))
310 b=INT(500+(10*RND(8))-(10*RND(8)))
320 d=2000
330 w=INT(10*RND(8)+95)
340 PRINT
350 PRINT"You have";a;" Vanders in the treasury."
360 PRINT INT(b);" peasants, ";
370 v9=INT(((RND(8)/2)*10+10))
380 IF c=0 THEN 400
390 PRINT INT(c);" foreign workers, ";
400 PRINT:PRINT"and";INT(d);" sq. miles of land."
410 PRINT"This year industry will pay";w;" Vanders":PRINT"for land......."
420 PRINT"Land currently costs";v9;" VanDers":PRINT" per sq mile to plant."
430 PRINT
440 PRINT"how many sq miles do you wish":INPUT "to sell to industry";h
450 IF h<0 THEN 440
460 IF h<=d-1000 THEN 520
470 PRINT" think again, you have only";d-1000;" sq miles"
```

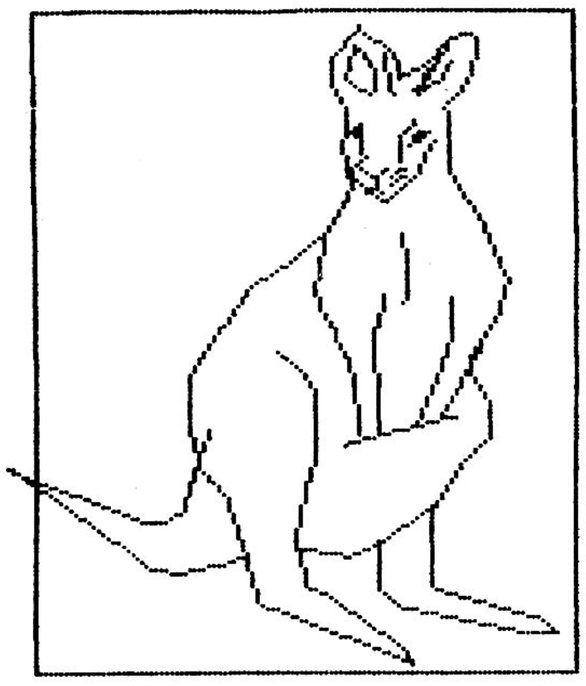
```
480 IF x<>0 THEN 440
490 PRINT"(Foreign industry will only":PRINT"buy farm land because forest":PRINT
"is uneconomical to strip":PRINT"mine due to trees e
tc.)"
500 x=1
510 GOTO 440
520 d=INT(d-h)
530 a=INT(a+(h*w))
540 PRINT"How many VanDers will you distribute":PRINT"to your peasants"
550 INPUT i
560 IF i<0 THEN 540
570 IF i<a THEN 650
580 IF i=a THEN 610
590 PRINT" think again, you have only";a;" VanDers"
600 GOTO 540
610 j=0
620 k=0
630 a=0
640 GOTO 970
650 a=INT(a-i)
660 PRINT"How many sq miles do you":PRINT"wish to plant";
670 INPUT j
680 IF j<0 THEN 660
690 IF j<=b*2 THEN 720
700 PRINT"Sorry, but each peasant can only":PRINT"plant 2 sq miles"
710 GOTO 660
720 IF j<=d-1000 THEN 750
730 PRINT"Sorry, but you've only";d-1000:PRINT"Sq Miles of farm land"
740 GOTO 660
750 u1=INT(j*v9)
760 IF u1<a THEN 830
770 IF u1=a THEN 800
780 PRINT"Think again, you've only";a:PRINT"VanDers left in the treasury."
790 GOTO 660
800 k=0
810 a=0
820 GOTO 970
830 a=a-u1
840 PRINT"How many VanDers do you wish to":PRINT"spend on pollution control?"
850 INPUT k
860 IF k<0 THEN 840
870 IF k<=a THEN 970
880 PRINT"Think again, you've only";a:PRINT"VanDers left in the treasury."
890 GOTO 840
900 IF h<>0 THEN 980
910 IF i<>0 THEN 980
920 IF j<>0 THEN 980
930 IF k<>0 THEN 980
940 PRINT
950 PRINT"Good bye..."
960 END
970 GOTO 900
980 PRINT
990 PRINT
1000 a=INT(a-k)
1010 a4=a
1020 IF INT(i/100)>=0 THEN 1050
1030 IF i/100<50 THEN 2070
1040 PRINT INT(b-(i/100));" peasants died of hunger!"
1050 f1=INT(RND(8)*(2000-d))
1060 IF k<25 THEN 1080
1070 f1=INT(f1/(k/25))
1080 IF f1<=0 THEN 1100
1090 PRINT f1;" peasants died of carbon":PRINT"monoxide and dust inhalation"
1100 IF INT((i/100)-b)<0 THEN 1130
1110 IF f1>0 THEN 1200
1120 GOTO 1210
1130 PRINT"You were forced to spend";INT(f1*9):PRINT"VanDers on Funeral expenses
"
1140 b5=f1
```

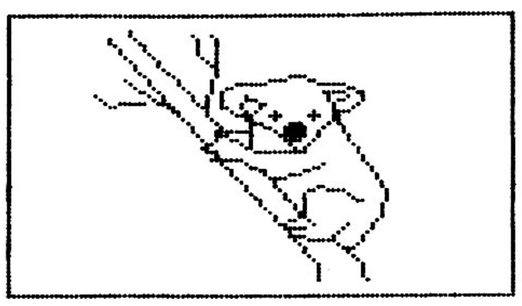
```
1150 a=INT(a-(f1*9))
1160 IF a>=9 THEN 1200
1170 PRINT"Not enough money to cover costs":PRINT"Land had to be sold"
1180 d=INT(d+(a/w))
1190 a=0
1200 b=INT(b-b5)
1210 IF h=0 THEN 1260
1220 c1=INT(h+(RND(8)*10)-(RND(8)*20))
1230 IF c>0 THEN 1250
1240 c1=c1+20
1250 PRINT c1;" workers came to the country":PRINT"and ";
1260 p1=INT(((i/100-b)/10)+(k/25)-((2000-d)/50)-(f1/2))
1270 PRINT ABS(p1);" peasants ";
1280 IF p1<0 THEN 1310
1290 PRINT"Came to "
1300 GOTO 1320
1310 PRINT"Left ";
1320 PRINT " Your country."
1330 b=INT(b+p1)
1340 c=INT(c+c1)
1350 u2=INT(((2000-d)*((RND(8)+1.5)/2)))
1360 IF c=0 THEN 1380
1370 PRINT" of";INT(j);" sq miles planted,"
1380 IF j>u2 THEN 1400
1390 u2=j
1400 PRINT" You harvested";INT(j-u2);" sq miles of crops."
1410 IF u2=0 THEN 1470
1420 IF t1>=2 THEN 1470
1430 PRINT"  (Due to ";
1440 IF t1=0 THEN 1460
1450 PRINT"increased ";
1460 PRINT"air and water pollution":PRINT"from foreign industry.)"
1470 q=INT((j-u2)*(w/2))
1480 PRINT" making";INT(q);" VanDers."
1490 a=INT(a+q)
1500 v1=INT(((b-p1)*22)+(RND(8)*500))
1510 v2=INT((2000-d)*15)
1520 PRINT" You made";ABS(INT(v1-v2));" VanDers from tourism."
1530 IF v2=0 THEN 1620
1540 IF v1-v2>=v3 THEN 1620
1550 PRINT"   decrease because ";
1560 g1=10*RND(8)
1570 IF g1<=2 THEN 1650
1580 IF g1<=4 THEN 1670
1590 IF g1<=6 THEN 1690
1600 IF g1<=8 THEN 1710
1610 IF g1<=10 THEN 1730
1620 v3=INT(a+v3)
1630 a=INT(a+v3)
1640 GOTO 1750
1650 PRINT"Fish population has dwindled":PRINT"due to water pollution."
1660 GOTO 1620
1670 PRINT"Air pollution is killing":PRINT"the birdie population!"
1680 GOTO 1620
1690 PRINT"mineral baths are being":PRINT"ruined by water pollution."
1700 GOTO 1620
1710 PRINT"unpleasant smog is dis-":PRINT"couraging sun bathers."
1720 GOTO 1620
1730 PRINT"Hotels are looking":PRINT"shabby due to smog grit."
1740 GOTO 1620
1750 IF b5>200 THEN 1920
1760 IF b<343 THEN 2070
1770 IF (a4/100)>5 THEN 2120
1780 IF c>b THEN 1570
1790 IF n5-1=x5 THEN 2170
1800 GOTO 2440
1810 PRINT"The number of foreign":PRINT"workers has exceeded the numnber"
1820 PRINT"of peasants. As a majority":PRINT"they have revolted and"
1830 PRINT"taken over the country."
1840 IF RND(8)<=0.5 THEN 1880
```

```
1850 PRINT"You have been thrown out":PRINT"of office and you are now"
1860 PRINT"sitting in a cold prison cell!"
1870 GOTO I890
1880 PRINT"You have been assassinated."
1890 PRINT
1900 PRINT
1910 PRINT"The End":END
1920 PRINT
1930 PRINT
1940 PRINT b5;" peasants died in 1 yr.!!!"
1950 PRINT"due to this extreme management":PRINT"You have not only"
1960 PRINT"been impeached and thrown":PRINT"out of office but you"
1970 m6=INT(RND(8)*10)
1980 IF m6<=3 THEN 2010
1990 IF m6<=6 THEN 2030
2000 IF m6<=10 THEN 2050
2010 PRINT"also had your eyes":PRINT"gouged out!!!!"
2020 GOTO 1890
2030 PRINT"Have also gained a":PRINT"really bad reputation!"
2040 GOTO 1890
2050 PRINT"Have also been ":PRINT"declared a Pac Wimp!!"
2060 GOTO 1890
2070 PRINT
2080 PRINT
2090 PRINT"Over one third of the":PRINT"population has died since":PRINT"you too
k office"
2100 PRINT"The people (remaining) hate":PRINT"your guts!!!!!"
2110 GOTO 1840
2120 IF b5-f1<2 THEN 1780
2130 PRINT
2140 PRINT"Money was left over":PRINT"in the treasury which":PRINT"you did not s
pend"
2150 PRINT" As a result, some of":PRINT"your peasants died of hunger.":PRINT"The
 remaining wallies":PRINT"decide that watching you d
ie":PRINT"will be good fun!"
2160 GOTO 1890
2170 PRINT"!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2180 PRINT"!! C O N G R A T U L A T I O N S !!"
2190 PRINT"!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
2200 PRINT"You have succesfully managed":PRINT"to complete a term of office!"
2210 PRINT"You were of course extremely":PRINT"lucky! But nevertheless, it's"
2220 PRINT"quite an achievement!":PRINT"Goodbye and good luck...Fascist!!"
2230 GOTO 1890
2240 PRINT"How many years had you ":PRINT"been in office before being ":PRINT"in
terrupted?"
2250 INPUT x5
2260 IF x5<0 THEN 1890
2270 IF x5<8 THEN 2300
2280 PRINT" come on, your term of":PRINT"office is only";n5;" yrs."
2290 GOTO 2240
2300 PRINT"how much did you have":PRINT"in the treasury"
2310 INPUT a
2320 IF a<0 THEN 1890
2330 PRINT"how many peasants"
2340 INPUT b
2350 IF b<0 THEN 1890
2360 PRINT"how many workers"
2370 INPUT c
2380 IF c<0 THEN 1890
2390 PRINT"how many sq. miles":PRINT"of land"
2400 INPUT d
2410 IF d<0 THEN 1890
2420 IF d>2000 THEN PRINT"Liar!!":GOTO 2390
2430 GOTO 330
2440 x5=x5+1
2450 b5=0
2460 GOTO 330
```

STAR WARS . . . By Jan Jacobsen

This is another one of the Aussie programs that has crossed my desk, and as usual it is of a really high standard. The object of the game is to blow everything up! Really neat! I won't say much more because the instructions are in the game itself! So enjoy this really neat program!





```
U
10 REM ************************************
20 REM ***                BY              ***
30 REM ***        JAN JACOBSEN            ***
40 REM ***           (C) 1985             ***
50 REM ************************************
60 SCREEN 2,2:COLOR1,15,,15:CLS
70 RESTORE
80 COLOR1,10,(0,0)-(255,191),10
90 COLOR1,6,(136,8)-(224,60),10
100 COLOR1,6,(130,168)-(255,191),10
110 LINE(130,168)-(255,191),1,B
120 LINE(143,80)-(247,152),1,B
130 LINE(136,8)-(231,60),1,B
140 COLOR1,15,(148,80)-(240,152),10
150 LINE(8,8)-(120,176),1,B
160 COLOR1,15,(8,8)-(115,176),10
170 CIRCLE(88,39),1,1,,,,BF
180 CIRCLE(75,38),1,1,,,,BF
190 CIRCLE(198,106),1,1,,,,BF
200 CIRCLE(206,106),1,1,,,,BF
210 CIRCLE(202,110),2,1,,,,BF
220 CURSOR134,170:PRINT"By Jan Jacobsen  (C)
230 CURSOR134,179:PRINT" FROM S.A USER CLUB"
240 R=R+1
250 READX,Y
260 IFX=0THENREADX,Y:LINE(X,Y)-(X,Y)
270 IFX=-1THEN530
280 LINE-(X,Y)
290 GOTO240
300 DATA0,0,76,12,70,20,71,28,73,32,73,34,72,40,73,44,74,50,72,56,68,64,60,72,52
,78,46,88,42,96,40,100,40,112,42,120,40,128,38,136,35,138,32,138
310 DATA 2,124,32,150,36,151,40,150,44,149,48,148,52,146,55,159,56,160,86,174,84
,170,80,166,68,156,64,154,62,145
320 DATA 72,147,79,143,80,156,84,160,108,166,116,166,112,162,90,154,90,136,92,13
2,96,124,102,116,102,112,102,108,100,104,98,100,98,96,100,92,102,88
330 DATA 104,84,105,80,106,76,104,72,100,60,94,48,94,40,93,32,97,28,99,24,100,20
,99,14,94,16,92,18,88,23,84,25,83,20,80,16,76,13,76,12
340 DATA 0,0,44,114,42,120,42,123,48,132,52,146,0,0,58,94,60,96,64,100,66,104,66
,108,66,120,64,128,62,140,62,145
350 DATA 0,0,75,17,74,20,73,24,76,28,78,29,79,28,79,24,77,20,75,17,0,0,81,19,82,
24,83,29
360 DATA 0,0,84,64,85,68,85,72,85,76,84,81
370 DATA 0,0,94,17,92,20,89,24,88,29,0,0,92,20,90,24,87,28,93,25,95,21,0,0,87,35
,84,39,84,45,0,0,76,36,77,44,74,47,74,49,80,55,88,48,91,40
```

```
380 DATA 0,0,81,47,79,49,79,51,79,49,76,47,0,0,77,53,78,52,79,52,80,53,0,0,90,38
,86,41,0,0,85,47,82,49,0,0,85,49,82,51
390 DATA 0,0,68,64,66,76,68,84,70,88,71,92,73,96,74,100,76,116,0,0,80,79,78,88,7
9,100,79,112,78,116,0,0,94,80,93,92,91,100,87,112,0,0,100,92,91,112,0,0,79,143,9
0,134,0,0,95,111,72,118,0,0
400 DATA 164,86,207,144,207,148,0,0,168,84,188,110,0,0,184,104,186,96,185,85,0,0
,186,94,182,90,181,85,0,0,178,105,167,97,0,0,171,103,165,104,161,101,0,0,196,120
,204,132,0,0,210,138,216,144,217,148,0,0
410 DATA 213,140,216,138,220,132,221,128,220,124,213,112,211,108,211,105,214,104
,216,102,216,100,212,98,209,98,204,96,200,97,196,98,192,98,187,100,187,104,193,1
08,193,114,0,0
420 DATA 187,111,192,114,194,115,204,115,210,107,210,104,208,100,212,100,213,101
,212,102,210,103,0,0
430 DATA 185,117,190,117,197,122,204,120,208,118,0,0,213,134,209,138,204,137,202
,137,0,0,216,128,212,125,208,124,204,126,204,132,206,134,204,132,202,133,200,134
,200,135,204,135,0,0
440 DATA 193,110,186,110,0,0,204,115,200,112,197,109,192,106,196,103,192,101,191
,103,193,106,0,0,186,111,184,114,183,114,184,115,186,115,0,0
450 DATA 140,12,140,28,144,28,144,24,148,28,152,28,148,20,152,16,152,12,144,18,1
44,12,140,12,0,0
460 DATA 156,12,156,28,160,28,160,24,164,24,164,28,168,28,168,12,156,12,0,0,160,
16,160,20,164,20,164,16,160,16,0,0
470 DATA 172,12,172,28,176,28,176,28,176,18,184,28,188,28,188,12,184,12,184,20,1
76,12,172,12,0,0
480 DATA 192,12,192,28,196,28,196,24,200,28,204,28,200,20,204,16,204,12,196,18,1
96,12,192,12,0,0
490 DATA 208,12,208,28,212,28,212,24,216,24,216,28,220,28,220,12,208,12,0,0,212,
16,212,20,216,20,216,16,212,16,0,0
500 DATA 144,36,144,48,156,48,156,52,144,52,144,56,160,56,160,44,148,44,148,40,1
60,40,160,36,144,36,0,0
510 DATA 164,36,164,56,180,56,180,36,164,36,0,0,168,40,168,52,176,52,176,40,168,
40,0,0
520 DATA 184,36,184,56,188,56,188,48,196,48,196,44,188,44,188,40,200,40,200,36,1
84,36,0,0
530 DATA 204,36,204,40,212,40,212,56,216,56,216,40,224,40,224,36,204,36,-1,-1
540 PAINT(141,13),1:PAINT(157,13),1:PAINT(173,13),1:PAINT(193,13),1:PAINT(209,13
),1
550 PAINT(145,37),1:PAINT(165,37),1:PAINT(185,37),1:PAINT(205,37),1
560 FORT=1TO500:NEXTT:GOTO570
570 CURSOR15,180:PRINT"INSTRU Y or N"
580 AN$=INKEY$:IFAN$=""THEN580
590 IF AN$="Y"THENBEEP:GOTO 1550
600 IF AN$="N"THENBEEP:GOTO 680
610 FORT=1TO100:NEXTT:GOTO570
620 REM ******************************
630 REM *      STAR WAR C 1985       *
640 REM *            BY              *
650 REM *      JAN JACOBSEN          *
660 REM *   ADELAIDE USER'S CLUB     *
670 REM ******************************
680 C=12:MAG1
690 GOSUB1470:GOSUB1100:GOSUB1290
700 A=0:XX=102:YY=80:X1=0:Y1=0:RN=0:G=0:SO=0:MU=16:SC=0:MI=0
710 Y1=0:X1=INT(RND(1)*100)+50
720 RN=INT(RND(1)*15)+1
730 SPRITE0,(XX,YY),0,4
740 ONRNGOTO750,760,750,750,760,760,750,760,750,760,750,760,750,750,770
750 SPRITE1,(X1,Y1),4,15:G=1:SO=6:GOTO 780
760 SPRITE1,(X1,Y1),7,8:G=2:SO=7:GOTO 780
770 SPRITE1,(X1,Y1),12,10:G=3:SO=10
780 IN$=INKEY$:IFIN$=""THEN860
790 IN=ASC(IN$)
800 IFIN$=" "THENSOUND5,1,15:GOTO 930:GOTO 780
810 IFIN=29ANDXX>17THENXX=XX-20
820 IFIN=30ANDYY>15THENYY=YY-20
830 IFIN=31ANDYY<145THENYY=YY+20
840 IFIN=28AND XX<172THENXX=XX+20
850 SPRITE0,(XX,YY),0,4
860 X2=INT(RND(1)*3)
870 IFX2=0THENX1=X1+SO
```

```
880 IFX2=1THENX1=X1-SO
890 Y1=Y1+SO
900 IFMI>=10THEN990
910 IFX1>=182ORX1<=160RY1>=160THENMI=MI+1:BLINE(40,180)-(70,190),1,BF:CURSOR 45,
182:COLOR11:PRINTMI:GOTO 710
920 ONGGOTO750,760,770
930 IF XX+8>=X1+4 AND XX+8<=X1+13 ANDYY+8>=Y1+4 AND YY+8<=Y1+13 THEN SPRITE1,(X1
,Y1),16,6:GOTO 950
940 SOUND5,1,0:GOTO810
950 SOUND4,0,15:FORA=0TO80:NEXT:SOUND0:IFSO=6THENSC=SC+30:GOTO980
960 IFSO=7THENSC=SC+80:GOTO980
970 IFSO=10THENSC=SC+200:GOTO980
980 BLINE(175,165)-(195,191),1,BF:CURSOR175,180:COLOR10:PRINTSC:GOTO710
990 CLS:SOUND0:CURSOR90,95:PRINT"YOUR SCORE";SC:CURSOR95,70:COLOR15:PRINT"GAME O
VER !"
1000 LINE(20,10)-(240,35),6,B
1010 CURSOR36,20:COLOR5:PRINTCHR$(17);"S T A R  W A R S  "
1020 PRINTCHR$(16)
1030 LINE(10,170)-(250,190),6,B
1040 COLOR14:CURSOR20,180:PRINT "KANKA Software (C)1985 By Jan Jacobsen"
1050 CURSOR52,130:PRINT"Do you want another game":COLOR7:CURSOR114,145:PRINT"(Y/
N)"
1060 IFINKEY$="Y"THENGOSUB 1290:GOTO700
1070 IFINKEY$="N"THEN1640
1080 GOTO1060
1090 RESTORE1480
1100 SCREEN2,2:COLOR,1,,1:CLS
1110 LINE(20,10)-(240,35),6,B
1120 LINE(10,170)-(250,190),6,B
1130 CURSOR36,20:COLOR5:PRINTCHR$(17);"S T A R  W A R S  "
1140 PRINTCHR$(16)
1150 SPRITE0,(50,70),4,15
1160 SPRITE1,(50,100),7,8
1170 SPRITE2,(50,130),12,10
1180 CURSOR80,78:COLOR15:PRINT"S C O U T     e 30 PTS"
1190 CURSOR80,108:COLOR7:PRINT"P A T R O L   e 80 PTS"
1200 CURSOR80,138:COLOR10:PRINT"F I G H T E R e 200 PTS"
1210 COLOR14:CURSOR20,180:PRINT "KANKA Software (C)1985 By Jan Jacobsen"
1220 RESTORE1260:FORJ=1TO110:READF:SOUND1,F,14:SOUND2,F+1,14:SOUND3,F*2,14:FORDE
=1TO30:NEXTDE,J
1230 READD
1240 IFD=0THENSOUND0:CLS:RETURN
1250 SOUND0,RETURN
1260 DATA 196,196,196,131,392,392,175,165,147,262,392,392,392,349,330,294,262,39
2,392,392,349,330,349,330,349,294,294,196,196,196,294,294,196,220,220,349,330,29
4,262,262,294,330,294,220,247,196,196,220,220,349,330,294,262,392,294,294,392,34
2,440,440,698
1270 DATA 698,659,587,523,523,587,659,587,440,494,392,392,523,466,415,392,349,31
1,294,262,392,784,196,196,196,262,392,392,349,330,294,523,392,392,392,349,330,29
4,523,392,392,392,349,330,349,294,294,294,294,0
1280 RETURN
1290 SCREEN2,2:COLOR,1:CLS
1300 FORP=0TO150:X=INT(RND(1)*200):Y=INT(RND(1)*170):E=INT(RND(1)*14)+2:PSET(X,Y
),E:NEXT
1310 CURSOR140,180:COLOR10:PRINT"SCORE":CURSOR10,180:COLOR10:PRINT"MISS"
1320 COLOR6:CURSOR215,5:PRINTCHR$(17);" S "
1330 COLOR15:CURSOR215,25:PRINT" T "
1340 COLOR4:CURSOR215,45:PRINT" A "
1350 COLOR7:CURSOR215,65:PRINT" R "
1360 COLOR14:CURSOR215,105:PRINT" W "
1370 COLOR10:CURSOR215,125:PRINT" A "
1380 COLOR6:CURSOR215,145:PRINT" R "
1390 PRINTCHR$(16)
1400 FORCI=1TO20STEP1
1410 CIRCLE(25,25),CI,11
1420 NEXTCI
1430 LINE(215,0)-(245,155),4,B
1440 LINE(210,0)-(250,155),4,B
1450 PAINT(211,1),4:PAINT(246,1),4
1460 RETURN
```

```
1470 RESTORE1500
1480 FORA=0TO19:READRE$:PATTERNS#A,RE$:NEXT:RETURN
1490 GOTO1480
1500 DATAF0F0C0C100010015,15000100C1C0F0F0,0F0F038300080000A8,A800800083030F0F
1510 DATA80C0E0A1E3A3E79E,9FE7A1E0A0E0C080,0103078SC7C5E779,F9E785070507030 1
1520 DATA0001021E3A57BAE7,0100000000000000,0080407SCEA5DE7,8000000000000000
1530 DATA0000000000205089,47A77FA344990403,0000000000040A91,E2E5FEC522992 0C0
1540 DATA43A3081C1E3F3F7F,FF7F7F3F3F1D3800,08183A70F2E1F8FE,FFFEF8ECC6829008
1550 SCREEN 1,1:CLS:PRINT "STAR WARS"
1560 PRINT"You are a member of the'FIGHTER SQUAD'
1570 PRINT "You are in your Fighter reday to blow up the EMPIRE Fighter"
1580 PRINT "The Fighter move randomly in a down   direction and are petty hard t
o catch.
1590 PRINT "IF you let more than ten EMPIRE       Fighter get away,you'll lose y
our job.
1600 PRINT "So hurry up!Jump into your space ship using the cursor key movements
 (up,   down,left and right),and speed to it!
1610 FORT=1TO 1500:NEXTT:PRINT :PRINT "Press any key to continue"
1620 IN$=INKEY$:IFIN$=""THEN1620
1630 GOTO 680
1640 SCREEN 1,1:CLS:PRINT "BYE":STOP
```

Russian Roulette

In this game, you are given by the computer a revolver loaded with one bullet and five empty chambers. You, spin the chamber and pull the trigger by inputting a "1", or if you want to give up then press "2". You win if you play ten times and your head is still in one peace.

```
1 PRINT "RUSSIAN ROULETTE"
3 PRINT :PRINT :PRINT
5 PRINT "THIS IS A GAME OF >>>>>>>> RUSSIAN ROULETTE"
10 PRINT "HERE IS THE REVOLVER...!"
20 PRINT "TYPE '1' TO SPIN CHAMBER AND PULL TRIGGER"
22 PRINT "OR '2' TO GIVE UP."
23 PRINT "OFF YOU GO..";
25 N=0
30 INPUT I
31 IF I<>2 THEN 35
32 PRINT "   CHICKEN!!!"
33 GOTO 72
35 N=N+1
40 IF RND(1)>.83333 THEN 70
45 IF N>10 THEN 80
50 PRINT "  -*-- CLICK --*-"
60 PRINT :GOTO 30
70 PRINT "     BANG!!!!! YOU'RE HEAD IS IN LOTS OF LITTLE BITS!"
71 PRINT "CONDOLENCES WILL BE SENT TO YOUR RELATIVES."
72 PRINT :PRINT :PRINT
75 PRINT ". . . NEXT VICTIM . . .":GOTO 20
80 PRINT "YOU WIN!!!"
85 PRINT "LET SOMEONE ELSE BLOW HIS BRAINS OUT!":GOTO 10
```

Awari

Michael Howard

Awai is an ancient African game played with seven sticks and thirty two stones or beans laid out as shown below:

```
                        My Side
          6     5     4     3     2     1
        ooo   ooo   ooo   ooo   ooo   ooo
My
Home                                        Your
        ooo   ooo   ooo   ooo   ooo   ooo  Home
          1     2     3     4     5     6
                       Your Side
```

The board is divided into six compartments or pits on each side. In addition, there are two special home pits at the ends.

A move is made by taking all of the beans from any (non-empty) pit on your own side. Starting from the pit to the right of this one, these beans are "sown" one in each pit working in each pit working in an anticlockwise direction.

A turn consists of one or two moves. If the last bean of your move is sown in your own home you may take a second move.

If the last bean sown in a move lands in an empty pit, provided that the opposite pit is not empty, all the beans in the opposite pit, together with the last bean sown are "captured" and moved to the players home.

When either side is empty, the game is finished. The player with the most beans in his home has won.

In the computer version, the board is printed as 14 numbers representing the 14 pits.

```
          3     3     3     3     3     3
    o                                       o
          3     3     3     3     3     3
```

The pits on your (lower) side are numbered 1-6 from left to right. The pits on my (the computer's) side are numbered from my left (your right!). Isn't this confusing?

To make a move you type in the number of a pit. If the last bean lands in your home, the Sega asks you if you want to go again, you, then enter your second move.

The computer's move is taken followed by a diagram of the board in its updated form. The computer always offers you the first move. This is actually an advantage for the computer!

This is a TRUE ARTIFICIAL INTELLIGENCE program as the computer actuall gets better at the game as the game continues, ie it learns from it's and your mistakes! And as it plays more games it gets to be just about impossible to beat!

```
10 REM MH
20 PRINT "AWARI
30 DATA 0
40 DIM B(13),G(13),F(50):READ N
50 PRINT :PRINT :E=0
60 FOR I=0 TO 12:B(I)=3:NEXT I
70 C=0:F(N)=0:B(13)=0:B(6)=0
80 GOSUB 350
90 PRINT "YOUR MOVE";:GOSUB 220
100 IF E=0 THEN 170
110 IF M=H THEN GOSUB 210
120 IF E=0 THEN 170
130 PRINT "MY MOVE IS ";:GOSUB 510
140 IF E=0 THEN 170
150 IF M=H THEN PRINT ",";:GOSUB 510
160 IF E>0 THEN 80
170 PRINT:PRINT "GAME OVER"
180 D=B(6)-B(13):IF D<0 THEN PRINT "I WIN BY";-D;" POINTS":GOTO 50
190 N=N+1:IF D=0 THEN PRINT "DRAWN GAME":GOTO 50
200 PRINT "YOU WIN BY";D;" POINTS":GOTO 50
210 PRINT "AGAIN..";
220 INPUT M:IF M<7 THEN IF M>0 THEN M=M-1:GOTO 240
230 SOUND 1,1000:PRINT "ILLEGAL MOVE":FOR QW=0 TO 100:NEXT:SOUND 0:GOTO 210
240 IF B(M)=0 THEN 230
250 H=6:GOSUB 270
```

```
260 GOTO 350
270 K=M:GOSUB 450
280 E=0:IF K>6 THEN K=K-7
290 C=C+1:IF C<9 THEN F(N)=F(N)*6+K
300 FOR I=0 TO 5:IF B(I)<>0 THEN 330
310 NEXT I
320 RETURN
330 FOR JI=7 TO 12:IF B(JI)<>0 THEN E=1:RETURN
340 NEXT ji
350 PRINT :PRINT "    ";
360 FOR I=12 TO 7 STEP -1:GOSUB 430
370 NEXT I
380 PRINT :I=13:GOSUB 430
390 PRINT "                              ";:PRINT B(6):PRINT "    ";
400 FOR I=0 TO 5:GOSUB 430
410 NEXT I
420 PRINT :PRINT :RETURN
430 IF B(I)<10 THEN PRINT " ";
440 PRINT B(I);:RETURN
450 P=B(M):B(M)=0
460 FOR P=P TO 1 STEP -1:M=M+1:IF M>13 THEN M=M-14
470 B(M)=B(M)+1:NEXT p
480 IF B(M)=1 THEN IF M<>6 THEN IF M<>13 THEN IF B(12-M)<>0 THEN 500
490 RETURN
500 B(H)=B(H)+B(12-M)+1:B(M)=0:B(12-M)=0:RETURN
510 D=-99:H=13
520 FOR I=0 TO 13:G(I)=B(I):NEXT I
530 FOR J=7 TO 12:IF B(J)=0 THEN 670
540 G=0:M=J:GOSUB 450
550 FOR I=0 TO 5:IF B(I)=0 THEN 600
560 L=B(I)+I:R=0
570 IF L>13 THEN L=L-14:R=1:GOTO 570
580 IF B(L)=0 THEN IF L<>6 THEN IF L<>13 THEN R=B(12-L)+R
590 IF R>Q THEN Q=R
600 NEXT I
610 Q=B(13)-B(6):IF C>8 THEN 650
620 K=J:  IF K>6 THEN K=K-7
630 FOR I=0 TO N-1:IF F(N)*6+K=INT(F(I)/6^(7-C)+0.1) THEN Q=Q-2
640 NEXT I
650 FOR I=0 TO 13:B(I)=G(I):NEXT I
660 IF Q>=D THEN A=J:D=Q
670 NEXT J
680 M=A:PRINT CHR$(42+M);:GOTO 270
690 FOR I=0 TO N-1:PRINT B(I):NEXT
```

Some more maths fun!

Try this program:-

```
10 REM MH
20 INPUT "Enter your house number";hn
30 an = 2 * hn + 5
40 an = an * 50
50 INPUT "Now enter your age";ag
60 an = an + ag
70 an = an + 365
80 an = an - 615
90 PRINT an
100 PRINT "Your house number is on the left, and your
    age is on the right!"
```

Pretty snazzy eh!?

SPRITE GENERATOR

We all know that the Sega has some of the easiest sprites to use in the business, so to help you even more at trying to grasp the immense power of the Sega's sprites, here is a program that just about everyone has been asking for, A sprite generator.

When you run the program, a box will be drawn and a small menu will appear on the right of the screen, with a little message "Please wait".

Here are the options open to you:-

Arrow keys . . . Move the "blob" about the box. If you go off the edge then the "blob" will appear on the other side of the box.

Space bar . . . This lets you fill in a small pixel in the 16 by 16 block. If the pixel is already filled in then when you press the space bar you erase that pixel.

Carriage return key . . . This lets you see the shape of the object on the hi-resolution screen, and then goes back to the text screen, note that in the top right hand corner there will appear 4 lots of data, this is the data for the sprite.

F . . . This lets you flip over to the hi-resolution screen to see your shape again, after you have created it.

Q . . . This lets you invert a line across, in other words, if a horizontal line is blank, then after you press "Q" that line will be filled. It's easier to see what I mean by experimenting.

W . . . As above but inverts a line downwards.

I . . . Inverts the whole 16 by 16 grid, thus creating an inverse of your sprite.

G . . . Flip the sprite across, a bit like looking at the shape in a mirrow.

H . . . As above but flips the shape upwards.

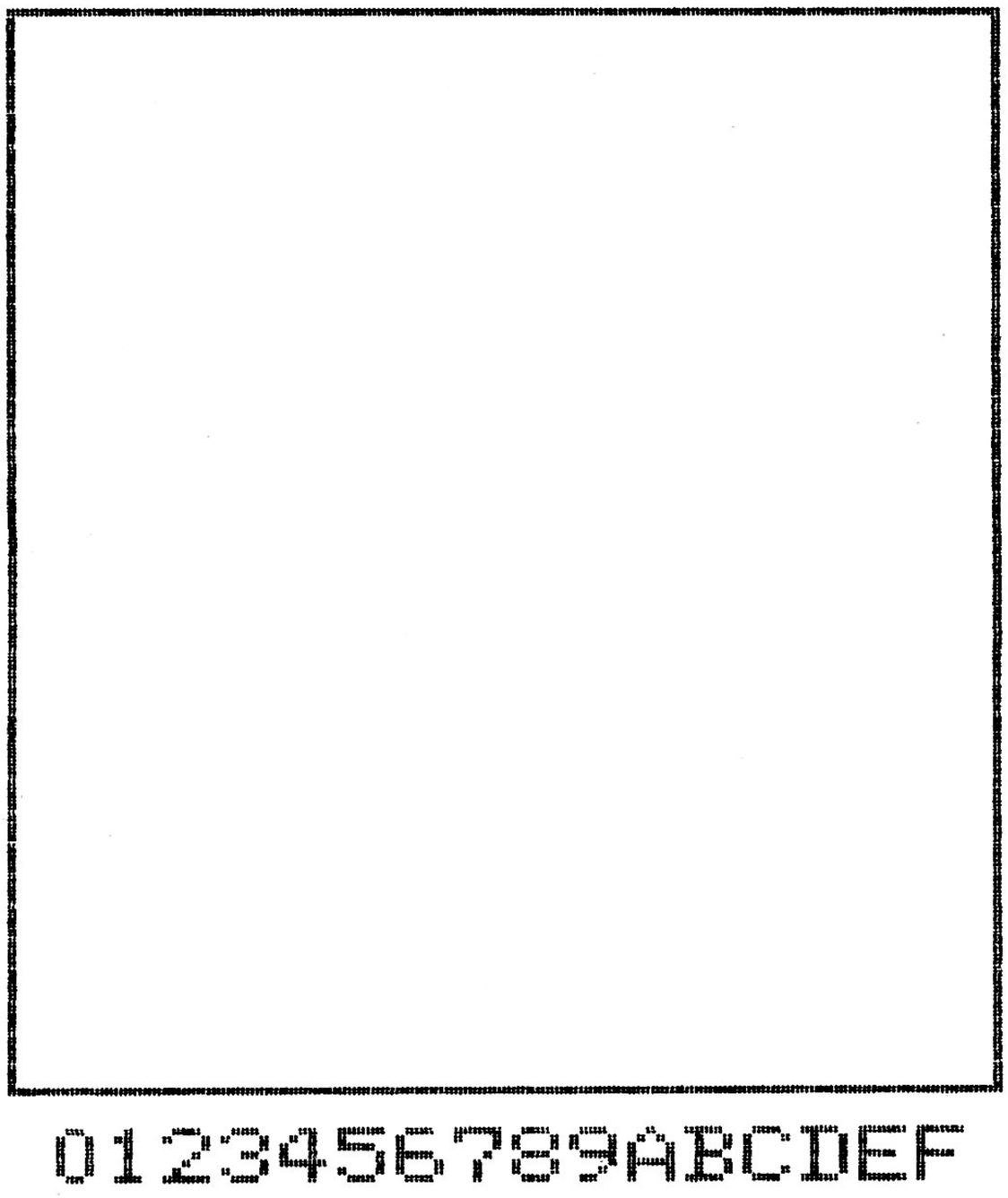
E,S,D,X . . . Lets you move the entire 16 by 16 matrix in any director, ie E moves the shape up, S left, D right and X is down. The object of these keys is to allow you to centre a shape.

Z . . . This allows you to enter hex or decimal data, so you can see what a shape looks like just from the data. Eg; if a sprite appears in a magazine with the following data 001E23FAFDCB0199 then enter &H00, &H1E etc and this will then draw that sprite. Hex has to start with ?H.

C . . . Lets you copy the data and shape of the sprite to the printer. The printer must be the SP-400 printer plotter.

L . . . Lets you fill in a shape. Fills a line at a time.

My only suggestion is to mess about with the controls until you get quite fluent in it. Oh, and by the way the "q","2","q","3","a" and "s" characters are in fact the characters that make up a box. Firstly press then "ENG DIERS" key, and then press the corresponding key, so where it says 2, just press 2, where it says q just press q, but make sure you, are using "ENG DIERS", you know that you are if the cursor is an asterisk "*;".




```
0 up,dn,lt,rt,spc
1 F,cr,R
2 Q-xor across
3 W-xor dwn
4 I-xor screen
5 G-flip across
6 H-flip down
7 E,S,D,X-rotate
8 Z-input hex/dec
9 C-Copy to Printer
A L-Line in bounds
B
C
D
E
F
```

SPRITE GENERATOR

```
1 REM MH
10 SCREEN 1,1:COLOR4,15:CLS:MAG1:F$=CHR$
(229)
20 PRINT"q2222222222222222w":FORA=1TO16:
PRINT"3                    3":HEX$(A-1):NEXT
:PRINT"a2222222222222222s",," 0123456789
ABCDEF":CURSOR20,7:PRINT"up,dn,lt,rt,spc
",TAB(20);"F,cr,R",TAB(20);"G-xor across
",TAB(20);"W-xor dwn",TAB(20);"I-xor scr
een",TAB(20);
30 PRINT"G-flip across",TAB(20);"H-flip
down",TAB(20);"E,S,D,X-rotate",TAB(20);"
Z-input hex/dec",TAB(20);"C-Copy to Prin
ter",TAB(20);"L-Line in bounds"
40 CURSOR3,9:PRINT"Please Wait";CHR$(19)
50 X=1:Y=1:ERASE:DIMA$(15,15),B$(15,15):
FORA=0TO15:FORB=0TO15:A$(A,B)="  ":NEXTB,
A:CURSOR3,9:PRINTSPC(13)
60 CURSORX,Y:PRINT" ":B$=A$(X-1,Y-1):CURS
ORX,Y:PRINTB$
70 C$=INKEY$:IFC$=""THEN60
80 SOUND1,1200,15:SOUND0
90 IFC$="R"THEN10
100 IFC$="Z"THENGOSUB600
110 IFC$="I"THEN530
120 IFC$="Q"THENGOSUB490
130 IFC$="W"THENGOSUB510
140 IFC$="G"THENGOSUB540
150 IFC$="S"THENGOSUB560
160 IFC$="D"THENGOSUB570
170 IFC$="E"THENGOSUB580
180 IFC$="X"THENGOSUB590
190 IFC$="H"THENGOSUB550
200 IFC$="C"THENGOSUB650
210 IFC$="F"THENGOSUB450
220 IFC$="L"THENGOSUB690
230 IFC$=CHR$(13)THEN340
240 IFC$="  "THENGOSUB320
250 X=X-(C$=CHR$(28))+(C$=CHR$(29))
260 Y=Y-(C$=CHR$(31))+(C$=CHR$(30))
270 IFX>16THENX=1
280 IFY>16THENY=1
290 IFY<1THENY=16
300 IFX<1THENX=16
310 GOTO60
320 IFA$(X-1,Y-1)="  "THENA$(X-1,Y-1)=F$:
RETURN
330 IFA$(X-1,Y-1)=F$THENA$(X-1,Y-1)="  ":
RETURN
340 DATA128,64,32,16,8,4,2,1
350 FORA=0TO3:Q$(A)="":NEXT
360 X=0:FA=0:TA=7:FB=0:TB=7:GOSUB400
370 X=2:FA=8:TA=15:FB=0:TB=7:GOSUB400
380 X=4:FA=0:TA=7:FB=8:TB=15:GOSUB400
390 X=6:FA=8:TA=15:FB=8:TB=15:GOSUB400:G
OTO430
400 X1=X/2:CURSOR20,X:FORA=FATOTA:T=0:RE
STORE:FORB=FBTOTB:READQ:IFA$(B,A)=F$THEN
T=T+Q
410 NEXTB:T$=HEX$(T):IFT<16THENT$="0"+T$
420 PRINTT$;:Q$(X1)=Q$(X1)+T$:NEXTA:PATT
ERNS#X1,Q$(X1):RETURN
430 SCREEN2,2:CLS:MAG1:SPRITE0,(128,96),
0,1
440 FORA=0TO1000:NEXT:SCREEN1,1:GOTO70
450 SCREEN 2,2:A=0:F=1:MAG1
460 IFA>75THENF=(3ANDF=1)+(1ANDF=3):MAGF
:A=0
```

```
470  A=A+1:IFINKEY$<>""THEN460
480  SCREEN1,1:RETURN
490  FORA=1TO16:CURSORA,Y:S$=F$:IFA$(A-1,
Y-1)=F$THENS$="  "
500  A$(A-1,Y-1)=S$:PRINTS$:NEXT:RETURN
510  FORA=1TO16:CURSORX,A:S$=F$:IFA$(X-1,
A-1)=F$THENS$="  "
520  A$(X-1,A-1)=S$:PRINTS$:NEXT:RETURN
530  Q=X:W=Y:FORY=1TO16:GOSUB490:NEXT:X=Q
:Y=W:GOTO60
540  FORA=0TO15:FORB=0TO15:B$(A,B)=A$(15-
A,B):NEXTB,A:FORA=0TO15:FORB=0TO15:A$(A,
B)=B$(A,B):CURSORA+1,B+1:PRINTA$(A,B):NE
XTB,A:RETURN
550  FORA=0TO15:FORB=0TO15:B$(A,B)=A$(A,1
5-B):NEXTB,A:FORA=0TO15:FORB=0TO15:A$(A,
B)=B$(A,B):CURSORA+1,B+1:PRINTA$(A,B):NE
XTB,A:RETURN
560  FORA=1TO15:A$(A-1,Y-1)=A$(A,Y-1):CUR
SORA,Y:PRINTA$(A-1,Y-1):NEXT:A$(15,Y-1)=
"  ":CURSOR16,Y:PRINT"  ":RETURN
570  FORA=15TO1STEP-1:A$(A,Y-1)=A$(A-1,Y-
1):CURSORA+1,Y:PRINTA$(A,Y-1):NEXT:A$(0,
Y-1)="  ":CURSOR1,Y:PRINT"  ":RETURN
580  FORA=1TO15:A$(X-1,A-1)=A$(X-1,A):CUR
SORX,A:PRINTA$(X-1,A):NEXT:A$(X-1,15)=""
":CURSORX,16:PRINT"  ":RETURN
590  FORA=15TO1STEP-1:A$(X-1,A)=A$(X-1,A-
1):CURSORX,A+1:PRINTA$(X-1,A):NEXT:A$(X-
1,0)="  ":CURSORX,1:PRINT"  ":RETURN
600  IFINKEY$<>""THEN600
610  CURSOR0,20:PRINT"Now enter upto 8 8-
bit data":FORA=1TO8
620  CURSOR0,21:PRINT"
                              ":CURSOR0,21:PRINT"Dat
a #";A;"  ";:INPUTQ:IFQ>255ORQ<0THEN620
630  RESTORE340:FORZ=1TO8:READD:IFQ-D>-1T
HENQ=Q-D:A$(Z-1,A-1)=F$:CURSORZ,A:PRINTF
$
640  NEXTZ,A:CURSOR0,20:PRINT"

                                    ":RETURN
650  CURSOR0,21:INPUT"Name of Sprite? ";Z
$:CURSOR0,21:PRINTSPC(20):LPRINTZ$:LPRIN
TCHR$(18);"S":LPRINT"A":FORA=0TO3:LPRINT
Q$(A):NEXT:LPRINTCHR$(18);"MO,-50":LPRIN
T"I"
660  FORA=0TO15:FORB=0TO15
670  IFA$(A,B)=F$THENLPRINT"M";A*3;",";40
-B*3:LPRINT"J2,0,0,2,-2,0,0,-2,1,1"
680  NEXTB,A:LPRINT"A":LPRINT:LPRINT:RETU
RN
690  REMIFX=1THEN720
700  FORA=XTO1STEP-1
710  IFA$(A-1,Y-1)=F$THEN730
720  A$(A-1,Y-1)=F$:CURSORA,Y:PRINTF$:NEX
T
730  F=0:IFX=16THENX=15:F=1
740  FORA=X+1TO16
750  IFA$(A-1,Y-1)=F$THEN770
760  A$(A-1,Y-1)=F$:CURSORA,Y:PRINTF$:NEX
T
770  IFFTHENX=16
780  RETURN
```

The Beginners Box of Bugs

I am having a few problems understanding the basic of BASIC, I know that this may seem a little stupid, but it just doesn't seem to make any sense!

BASIC has become the most widely used programming language this side of Pluto! And as such, is an important tool for all computer users.

The most effective way of learning a programming language is through actual practice. This little tutorial has been designed to teach BASIC through a few simple graduated exercises. It is written for all readers who have a minimum scientific or technical background and who want to learn through actual experience, by studying realistic examples, how to program in Sega BASIC.

Anyone can learn to program in BASIC, by working through some practical exercises. This tutorial will demonstrate that programming is not just for professionals. Starting with a simple exercise you will be taught the rudimentry instructions, what they do and how to use these with the rules of BASIC. No prior knowledge of BASIC is needed.

The reason for this little bit coming about is due to the inadequte little blue book that comes with the Sega, it is a good reference manual, but lousy to learn from!

First Program . . . A Simple Equation

Imagine the following equation:-

Weight in Pounds = 14 x Weight in Stones

This simple program converts weight in stones to weight in pounds, by multiplying stones by 14.

The program to do this may look like this:-

```
10 INPUT S          Read in weight in stones
20 P=14*S           Calculate weight in pounds
30 PRINT P          Print out the result
40 END
```

Although simple, this program brings up several points about the format of a BASIC program:

— Each line has a line number
— Each line carries an instruction
— The read instruction, ie., the **INPUT** instruction, is used to get information into the computer.
— In the instruction on line 20 multiplication is represented by an asterisk.
— The program is terminated by an **END** instruction, but this is optional.

To enter the above program just type out the line number, then a space and then type in the instructions on that line. When you have finished typing in the data for that line press the **CR** key (this is the big one just left of the four arrow/cursor keys), then type in the next line. When you have typed in all the program lines type in **RUN** and then press the **CR** key.

When the program is run, you will see a **?** on the screen. This question mark means that the computer is waiting for some information. In this case it is expecting a number that is a weight in stones, as an example enter the number 11, in other words 11 stones in weight. Now pres the **Cr** key.

Note that when you want to give the computer some information you must press the **CR** key after you have typed in the data. In this case after you have typed in 11 you press the **CR** key.

Okay, now that you have typed in the data and press the **CR** key the computer will chew on the info and spit out the number 154. In other words 11 times 14 is 154, 11 stones is equal to 154 pounds.

So what has the program done? Well to start off, line 10 has prompted the user (that's you!) to input a number.

Line 20 then works out the value of your input into pounds of weight.

Line 30 then prints out the value of your weight in pounds.

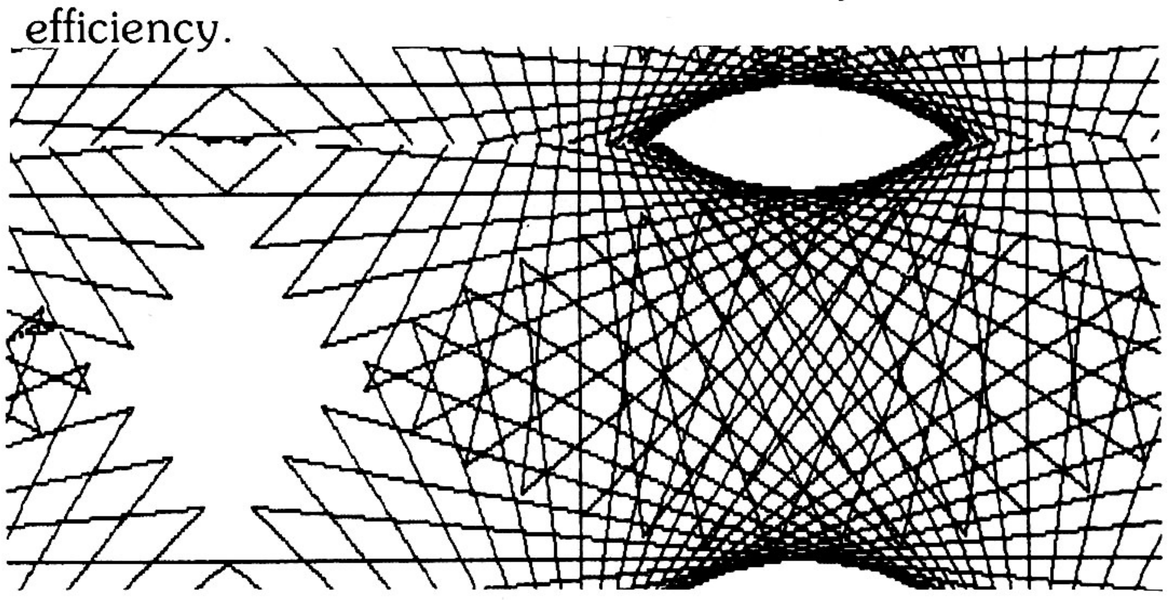
The last line tells the computer to stop, this line is optional, but most books on programming reckon that you must always put an **END** in for good measure, so to keep in line with these books, I've put in an **END!**

As the program stands, it is mathematically correct, the major problem is that it is about as **user friendly** as a brick! The words **User Friendly** mean just that . . . a program that is user friendly is easy to use from the user's point of view (don't forget a user is a person like yourself). I'll now show you a version of the same program that is a little more user friendly.

Firstly, type in **NEW** and then press the **CR** key. NEW wipes any program that is in the computer's memory. Now enter the following program:-

```
10 CLS
20 PRINT "A Program to convert stones": PRINT "to
   pounds (weight)"
30 PRINT:PRINT
40 INPUT "Enter a weight in stones";S
50 P=14*S
60 PRINT:PRINT
70 PRINT S;"stones is equal to ";P;"pounds"
80 END
```

Now, this program may appear to be a little bit too complex for the job in hand, well maybe it is, but that does not matter as this is a tutorial in style more than efficiency.



Line 10 clears the screen, this prevents the screen from getting too cluttered. A jumbled up screen is a mess!

Line 20 tells the user what the program does . . . very important!

Line 30 prints two spaces, thus separating the next lot of text from the title, once more this aids in making the screen more readable.

Line 40 actually serves as two lines. It is the same as:-

```
40 PRINT "Enter a weight in stones";
41 INPUT S
```

Luckily, the BASIC on the SEGA let's you group together a PRINT-PRINT sequenace, study line 40 a little more, and you'll see how it works. S will hold the value of the weight in stones.

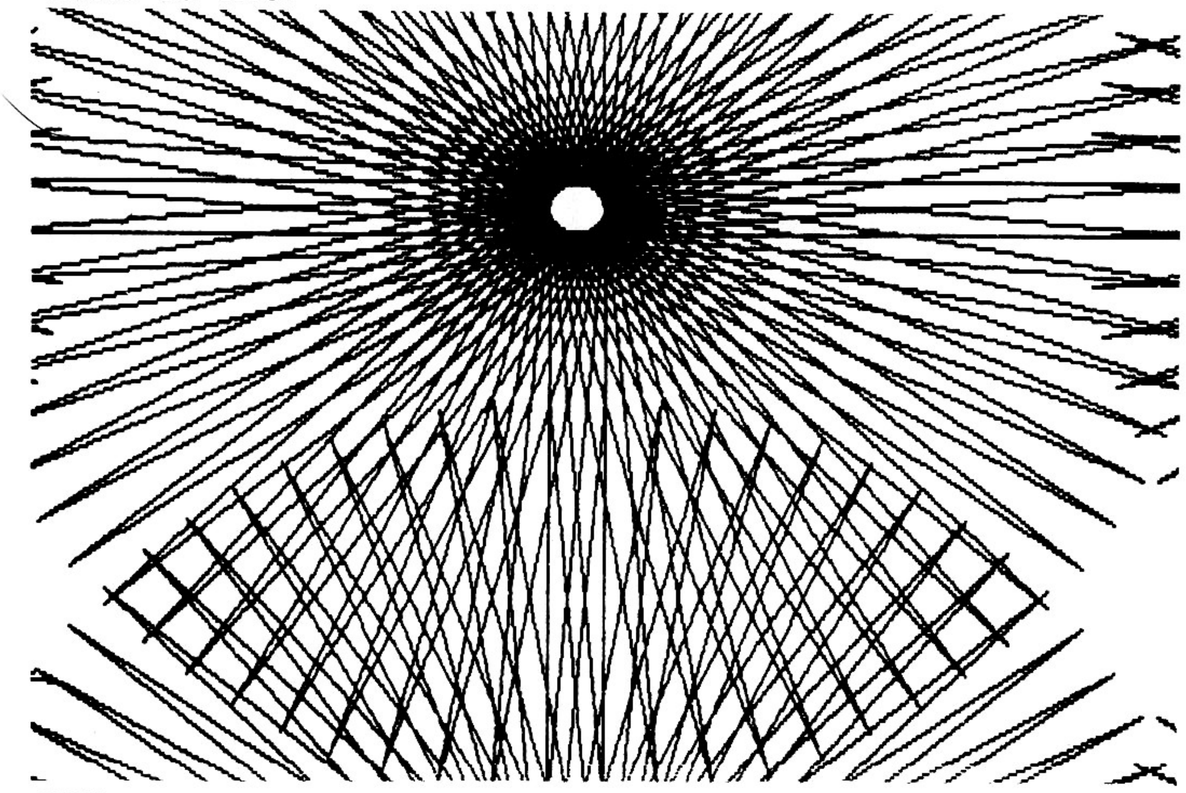
Line 50 this converts the stones into pounds, by multiplying **S** by **14, the variable P** is then assigned the value of S times 14. Note that the * symbol is used to multiply, + , − and /, add, subrtact and divide respectively, but * is a little odd and serves as multiply on 99.99% of all computers. P will now hold the value of the weight in pounds.

Line 60 separates the screen a little, aiding in clarity.

Line 70 this prints out the final result. Note how variables, in this case S and P, are separated from text, (the stuff inside "'s) by colons (;). This is extremely important. To sum up all data must be separated from text by **;.OKAY?!!!**

Line 80 The End

Well that is hopefully going to clear up a few problems that a few people are having. For any more help please write to me.



Why are most BASIC's different, I thought BASIC was BASIC and that's that!

There is no such thing as a standard BASIC. Or rather there are a number of standard BASIC's. By dint of it's popularity, the most standard of BASIC's is MicroSoft BASIC. On the other hand the standard BASIC in educational institutions in Britain is BBC BASIC. Here is a quick look at some BASIC's available on micro computers.

Microsoft BASIC is the best known dialect of the language. Commonly known as MBASIC, it is the granddaddy of nearly all home computer BASIC's. It is used in business machines such as the IBC PC and the ACT Apricot. There is a special version of the language called version 2.0 that runs on the Apple Macintosh and makes use of it's more advanced features. More than anything the Mac is a graphics machine and the version of MBASIC that it uses is tailored to meet those needs.

MSX BASIC is directly derived from the original Microsoft dialect, as are the versions used by the Commodore micros-though Microsoft disown the awful implementation on the Commodore 64 and 128!

CBASIC is Digital Research's answer to Microsoft BASIC and can be found on a number of CP/M machines. It's special features include the ability to split a program into lots of little "lumps" or procedures thus enabling programs to be debugged a lot quicker.

ANSI BASIC is an attempt to standardize BASIC. It took the ANSI (American National Standards Institute) the worst part of ten years to define the language, and although it is very good, the world as a whole doesn't know that it exists! You can find a version of the BASIC running on a computer called the Enterprise, and it's BASIC is called IS-BASIC, or Intelligent Software BASIC.

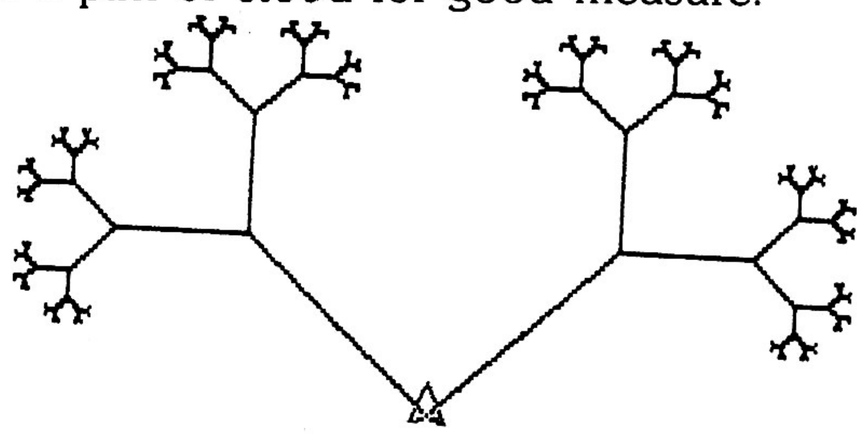
GW-BASIC is known as "Gee Whizz" BASIC — apparently the Americans liked it when they first saw it! Underneath all the razzzmatazzz, GWBASIC is Microsoft BASIC with a few frills.

Big computers have big BASIC's. Hewlett-Packards BASIC is over 200k long! But it is designed for scientific number crunching, and has a whole host of built-in functions for complex math. DEC BASIC Plus is similar, and runs on $5000 plus machines.

Sega BASIC has one point that kills all other BASIC's and that is the editor. If you, make a boo boo in a program it is a simple matter to correct that line, you, just move the cursor up to the line and alter it . . . simple as that! On, say a ZX-Spectrum, you, have to call up the offending line. Move the cursor to the error, which can take up to 1 minute to get to the end of a line! And then alter it . . . very time consuming. The Sega's editor is the best I have **EVER** seen!

Amstard BASIC is very powerful and fast, it supports graphics and sound very well. It also lets the user run up to 5 small sub programs at once! It's BASIC does lack a few Disc operating commands though.

On the whole, BASIC is not a portable language. That is, programs written without a few modifications. However right at the top of the heap is a BASIC called BLS, BASIC Language System, this, in theory let's the programmer write a BASIC program, and then later on, that program can be made to run on any machine . . . there is one slight draw back . . . cost, to buy BLS you more or less have to pay an arm and a leg and possibly throw in a pint of blood for good measure!



Why should a **BASIC** programmer learn Logo as a second language? What programs can be written in Logo that can't be written in BASIC?

The short answer to this is, none; but Logo is better suited to some types of progam than BASIC is, and enables you to write them more quickly and economically. Perhaps more importantly, though, is the difference in thought processes between programming in the two languages.

It's the fact that you have to think differently when you progam in Logo that underlies it's use as an educational language. Many home computer versions of Logo are designed for education use, following the guidelines put down by it's inventor Seymour Papert, and most have good manuals.

To a BASIC programmer, it can be difficult to realise how many elements of the language are optional. What! No line numbers? What! A language that need not have a program to run? Both these are true of Logo.

In Logo, you write short procedures, procedures which use other procedures which use other procedures and so on and so on. You don't use a special thing called a program, you run one of these procedures.

From another angle, you might think of each Logo procedure as the definition of a word in the Logo language. Words which you define yourself (like Forth, covered later in the magazine) — say SQUARE, or CIRCLE-once their definitions have been stored in memory, they can be used as though they were standard Logo commands.

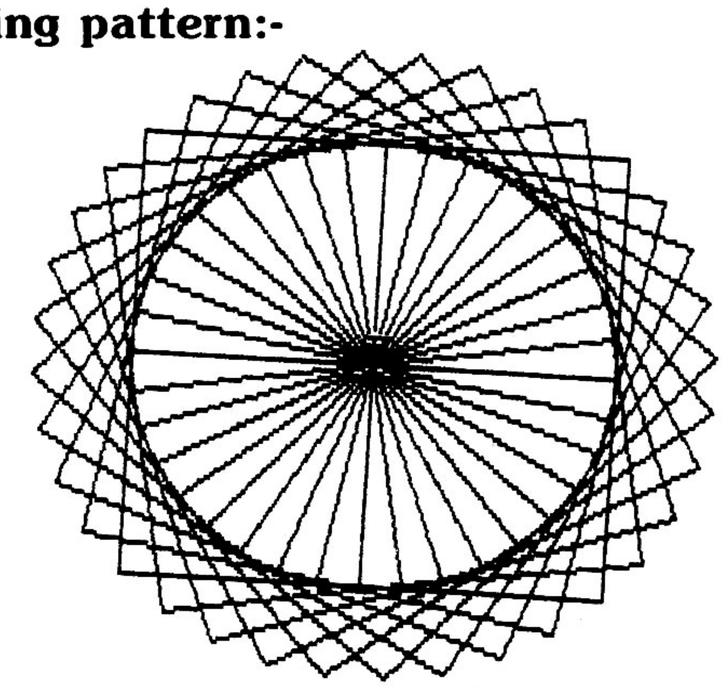
Logo does in fact go beyond graphics, it can be used to create very powerful data bases, powerful enough to be used in artificial intelligence work. It is similar to other languages, such as LISP and Prolog in this respect.

One of the very best Logo's is available, on disc for the Sega and disc drive combination, I can only say one thing, if you have a disc drive and want to learn a new and innovative language, then purchase Logo!

Here is an example program that draws a while host of boxes on the screen:-

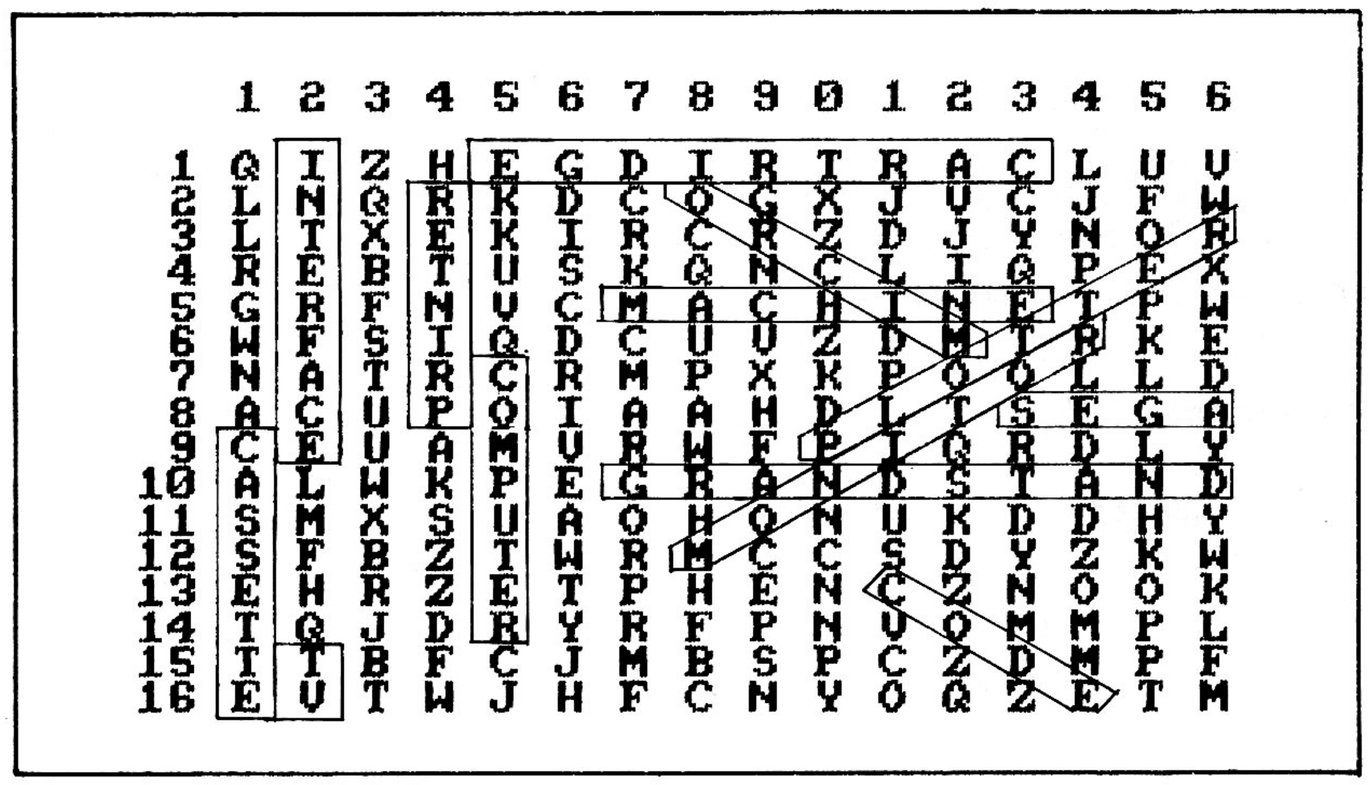
```
TO BOX
  REPEAT 4 [FD 100 RT 90]
END

TO BOXES
  REPEAT 36 [RT 10 BOX]
END
```

If, when in logo, you type Boxes, you'll get the following pattern:-



Competition . . .
The Solution

Well, here is the solution to the word search:-


1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6
1 Q I Z H E C D I R T R A C L U V
2 L I N T E R F A C E L M F H Q I
3 L R G W N A C A S S E T T E U W
4 R F A C E L M F H Q I V R X W
5 G W N A C A S S E T T E D A Y
6 A C A S S E T T E L E D A Y D
7 S S E T T E L M F H Q I V Y W
8 E T T E L M F H Q I V K L F M
9 T E L M F H Q I V K L F M J T
10 E L M F H Q I V K L F M J T Z
11 L M F H Q I V K L F M J T Z Q
12 M F H Q I V K L F M J T Z Q X
13 F H Q I V K L F M J T Z Q X B
14 H Q I V K L F M J T Z Q X B F
15 Q I V K L F M J T Z Q X B F S
16 I V K L F M J T Z Q X B F S T

Basketball

In this program you play the captain of the Sega College Basketball team, and the program simulates the game really well.

You are, as I have said, the captain of the Sega team and you can control the type of shot and defence that you use through out the game.

There are four basic types of shots:

1) Long Jump Shot. (30ft)
2) Short Jump Shot. (15ft)
3) Lay Up.
4) Set Shot.

Both teams use the same defence, but you may choose it by pressing:

6) Press defence
6.5) Man to man defence
7) Zone defence
7.5) No defence

To change the defence press 0 as your next shot.

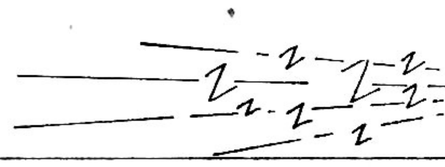
Note that the game is biased ever so slightly towards Sega College. The average probability of a Sega shot being good is 62.95% compared to a probability of 61.85% for the opponents (the computer).

```
10 REM MH
20 CLS
30 PRINT"Basketball"
40 PRINT"In this game you will play"
50 PRINT"The part of Sega College"
60 PRINT"& play against some other"
70 PRINT"scummy team eg; Scumbag College"
80 PRINT"The Young Ones fans please take note!"
90 PRINT:PRINT
100 INPUT "What is your starting defence ";d
110 INPUT"Please name your opponents ";o$
120 p=1
130 PRINT"Centre jump"
140 IF RND(8)> 3/5 THEN 170
150 PRINT o$;" controls the tap."
160 GOTO 1220
170 PRINT"Sega College controls the tap."
180 PRINT
190 INPUT"Your shot";z
200 p=0
210 IF z<>INT(z) THEN 240
220 IF z<0 OR z>4 THEN 240
230 GOTO 250
240 PRINT"Incorrect answer. Please retype ";:GOTO 190
250 IF RND(8)<0.5 THEN 430
260 IF t<100 THEN 430
270 PRINT
280 IF s(1)<>s(0) THEN 360
290 PRINT" *** End of Second Half *** "
300 PRINT"Score at end of regulation time:"
310 PRINT"          Sega";s(1);" ";o$;s(0)
320 PRINT
330 PRINT"Begin two minute overtime period"
340 t=93
350 GOTO 130
360 PRINT"  **** End of the Game **** "
370 PRINT"Final Score: Sega";s(1);" ";o$;s(0)
380 STOP
390 PRINT
400 PRINT"*** Two minutes left in the game ***"
```

```
410 PRINT
420 RETURN
430 ON z GOTO 450,450
440 GOTO 860
450 T=T+1
460 IF t=50 THEN 1900
470 IF t=92 THEN 490
480 GOTO 500
490 GOSUB 390
500 PRINT"Jump Shot"
510 IF RND(8)>0.341*d/8 THEN 550
520 PRINT"Shot is good."
530 GOSUB 1870
540 GOTO 1220
550 IF RND(8)>0.682*d/8 THEN 730
560 PRINT"Shot is off target."
570 IF d/6*RND(8)>0.45 THEN 600
580 PRINT"Sega controls the rebound."
590 GOTO 620
600 PRINT"Rebound to ";o$
610 GOTO 1220
620 IF RND(8)>0.4 THEN 640
630 GOTO 860
640 IF d=6 THEN 1790
650 PRINT"Ball passed back to you. ";
660 GOTO 190
670 IF RND(8)>0.9 THEN 710
680 PRINT"Player fouled, two shots."
690 GOSUB 1620
700 GOTO 1220
710 PRINT"Ball stolen. ";o$;"'s ball."
720 GOTO 1220
730 IF RND(8)>0.782*d/8 THEN 800
740 PRINT"Shot is blocked. Ball controlled by..";
750 IF RND(8)>0.5 THEN 780
760 PRINT"Sega."
770 GOTO 190
780 PRINT o$;"."
790 GOTO 1220
800 IF RND(8)>0.843*d/8 THEN 840
810 PRINT"Shooter fouled. Two shots."
820 GOTO 1620
830 GOTO 1220
840 PRINT"Charging foul. Sega loses ball."
850 GOTO 1220
860 t=t+1
870 IF t=50 THEN 1900
880 IF t=92 THEN 900
890 GOTO 910
900 GOSUB 390
910 IF z=0 THEN 1190
920 IF z>3 THEN 1170
930 PRINT"Lay up."
940 IF 7/d*RND(8)>0.4 THEN 980
950 PRINT"Shot is good. Two points."
960 GOSUB 1870
970 GOTO 1220
980 IF 7/d*RND(8)>0.7 THEN 1080
990 PRINT"Shot is off the rim!"
1000 IF RND(8)>2/3 THEN 1030
1010 PRINT o$;" controls the rebound."
1020 GOTO 1220
1030 PRINT"Sega controls the rebound."
1040 IF RND(8)>0.4 THEN 1060
```

```
1050 GOTO 860
1060 PRINT"Ball passed back to you.";
1070 GOTO 190
1080 IF 7/d*RND(8)>0.875 THEN 1120
1090 PRINT"Shooter fouled. Two shots."
1100 GOSUB 1620
1110 GOTO 1220
1120 IF 7/d*RND(8)>0.925 THEN 1150
1130 PRINT"Shot blocked. ";o$;"'s ball."
1140 GOTO 1220
1150 PRINT"Charging foul. Sega loses the ball."
1160 GOTO 1220
1170 PRINT"Set shot."
1180 GOTO 940
1190 INPUT"Your new defensive alignment is ";d
1200 IF d>6 THEN 1190
1210 GOTO 180
1220 p=1
1230 t=t+1
1240 IF t=50 THEN 1900
1250 GOTO 1270
1260 GOSUB 390
1270 PRINT
1280 z1=10/4*RND(8)+1
1290 IF z1>2 THEN 1520
1300 PRINT"Jump shot."
1310 IF 8/d*RND(8)>0.35 THEN 1350
1320 PRINT"Shot is good."
1330 GOSUB 1840
1340 GOTO 180
1350 IF 8/d*RND(8)>0.75 THEN 1460
1360 PRINT"Shot is off the rim."
1370 IF d/6*RND(8)>0.5 THEN 1400
1380 PRINT"Sega controls the rebound."
1390 GOTO 180
1400 PRINT o$;" controls the rebound."
1410 IF d=6 THEN 1740
1420 IF RND(8)>0.5 THEN 1450
1430 PRINT"Pass back to ";o$;" guard."
1440 GOTO 1220
1450 GOTO 1520
1460 IF 8/d*RND(8)>0.9 THEN 1500
1470 PRINT"Player fouled. Two shots."
1480 GOSUB 1620
1490 GOTO 180
1500 PRINT"Offensive foul. Sega's ball."
1510 GOTO 180
1520 IF z1>3 THEN 1600
1530 PRINT"Lay up."
1540 IF 7/d*RND(8)>0.413 THEN 1580
1550 PRINT"Shot is good."
1560 GOSUB 1840
1570 GOTO 180
1580 PRINT"Shot is missed."
1590 GOTO 1370
1600 PRINT"Set shot."
1610 GOTO 1540
1620 REM
1630 IF RND(8)>0.49 THEN 1680
1640 PRINT"Shooter makes both shots."
1650 s(1-p)=s(1-p)+2
1660 GOSUB 1850
1670 RETURN
1680 IF RND(8)>0.75 THEN 1720
```

```
1690 PRINT"Shooter makes 1 shot & misses."
1700 s(1-p)=s(1-p)+1
1710 GOTO 1660
1720 PRINT"Both shots missed!"
1730 GOTO 1660
1740 IF RND(8)>0.75 THEN 1760
1750 GOTO 1420
1760 PRINT"Ball stolen. Easy lay up for Sega."
1770 GOSUB 1870
1780 GOSUB 1220
1790 IF RND(8)>0.6 THEN 1810
1800 GOTO 650
1810 PRINT"Pass stolen by ";o$;" easy layup."
1820 GOSUB 1840
1830 GOTO 180
1840 s(0)=s(0)+2
1850 PRINT"Score: ";s(1);" to";s(0)
1860 RETURN
1870 s(1)=s(1)+2
1880 GOSUB 1850
1890 RETURN
1900 PRINT"*** End of the first half ***"
1910 PRINT"Score: Sega ";s(1);" ";o$;s(0)
1920 PRINT:PRINT:GOTO 130
```



Buzzword Generator

This program is invaluable for preparing speeches and briefings about educational technology. This buzzword generator provides sets of three highly acceptable words to work into your material. Your audience will think that you must be some mega brain as you verbally throw up this garbage! And they will never know that the phrases don't actually mean anything because they sound so great!

To run the program just read through the program.

```
10 CLS
20 PRINT"Buzzword generator"
30 PRINT"This program prints out highly"
40 PRINT"acceptable phrases for use in"
50 PRINT"Speeches etc."
60 PRINT:PRINT
70 PRINT"When a phrase is made, just"
80 PRINT"press 'Y' to create another"
90 PRINT:PRINT
100 PRINT"Hers's the first phrase.."
110 DIM a$(40)
120 FOR i=1 TO 39:READ a$(i):NEXT i
130 PRINT a$(INT(RND(9)*13)+1);" ";
140 PRINT a$(13*RND(8)+14);" ";
150 PRINT a$(13*RND(8)+27):PRINT:PRINT
160 a$=INKEY$:IF a$="" THEN 160
170 IF a$="y" OR a$="Y" THEN 130
180 PRINT:PRINT"Come back again sometime!"
190 END
200 DATA ability, basal, behavioral, child-centered
210 DATA differentiated, discovery, flexible, heterogeneous
220 DATA homogenous, manipulate, modular, tavistock
230 DATA individualised, learning, evaluative, objective
240 DATA cognitive, enrichment, scheduling, humanistic
250 DATA integrated, non-graded, training, vertical age
260 DATA motivational, creative, grouping, modification
270 DATA accountability, process, core curriculum, algorithm
280 DATA performance, reinforcement, open classroom, resource
290 DATA structure, facility, enviroment
```

3D Noughts and Crosses

3D Tic-Tac-Toe, call it whatever you like, is the game of normal Tic-Tac-Toe on a 4x4x4 cube. You must get 4 markers in a row or diagonal along any 3-dimensional plane in order to win.

Each move is indicated by a 3 digit number (digits not separated by commas), with each digit between 1 and 4 inclusive.

The digits indicate the level, column and row respectively, of the move. You can win at this game, but only if you play really well, as the game is considerably more difficult to plat than conventional 3x3 Noughts and Crosses.

```
10 REM MH
20 SCREEN 1,1:CLS
30 PRINT"3D Noughts and Crosses"
40 DIM x(64),l(76),m(76,4),y(16)
50 FOR i=1 TO 16
60 READ y(i)
70 NEXT i
80 FOR i=1 TO 76
90 FOR j=1 TO 4
100 READ m(i,j)
110 NEXT j
120 NEXT i
130 FOR i=1 TO 64
140 x(i)=0
150 NEXT i
160 z=1
170 PRINT"Do you want to move first "
180 s$=INKEY$:IF s$="" THEN 180
190 IF s$="y" OR s$="Y" THEN 220
200 IF s$="n" OR s$="N" THEN 410
210 GOTO 180
220 GOSUB 2300
230 PRINT"your move";
240 INPUT j1
250 IF j1=1 THEN 2510
260 IF j1<>0 THEN 290
270 GOSUB 2300
280 GOTO 220
290 IF j1<111 THEN 2500
300 IF j1>444 THEN 2500
310 GOSUB 2250
320 k1=INT(j1/100)
330 j2=(j1-k1*100)
340 k2=INT(j2/10)
350 k3=j1-k1*100-k2*10
360 m=16*k1+4*k2+k3-20
370 IF x(m)=0 THEN 400
380 PRINT"That square is occupied."
390 GOTO 220
400 x(m)=1
410 GOSUB 1400
420 j=1
430 i=1
440 IF j=1 THEN 500
450 IF j=2 THEN 570
460 IF j=3 THEN 710
470 i=i+1:IF i<=76 THEN 440
480 j=j+1:IF j<=3 THEN 430
490 GOTO 1080
500 IF l(i)<>4 THEN 470
510 PRINT"You win as follows";
520 FOR j=1 TO 4
530 m=m(i,j)
540 GOSUB 1330
```

```
550 NEXT j
560 GOTO 1260
570 IF l(i)<>15 THEN 470
580 FOR j=1 TO 4
590 m=m(i,j)
600 IF x(m)<>0 THEN 640
610 x(m)=5
620 PRINT"Machine moves to ";
630 GOSUB 1330
640 NEXT j
650 PRINT", and wins as follows."
660 FOR j=1 TO 4
670 m=m(i,j)
680 GOSUB 1330
690 NEXT j
700 GOTO 1260
710 IF l(i)<>3 THEN 470
720 PRINT"Nice try. Machine moves to";
730 FOR j=1 TO 4
740 m=m(i,j)
750 IF x(m)<>0 THEN 790
760 x(m)=5
770 GOSUB 1330
780 GOTO 220
790 NEXT j
800 GOTO 1080
810 i=1
820 l(i)=x(m(i,1))+x(m(i,2))+x(m(i,3))+x(m(i,4))
830 l=l(i)
840 IF l<2 THEN 910
850 IF l>=3 THEN 910
860 IF l>2 THEN 1980
870 FOR j=1 TO 4
880 IF x(m(i,j))<>0 THEN 900
890 x(m(i,j))=1/8
900 NEXT j
910 i=i+1:IF i<=76 THEN 820
920 GOSUB 1400
930 i=1
940 IF l(i)=1/2 THEN 2110
950 IF l(i)=1+3/8 THEN 2110
960 i=i+1: IF i<=76 THEN 940
970 GOTO 1590
980 z=1
990 IF x(y(z))=0 THEN 1030
1000 z=z+1
1010 IF z<>17 THEN 990
1020 GOTO 1480
1030 m=y(z)
1040 x(m)=5
1050 PRINT"Machine moves to";
1060 GOSUB 1330
1070 GOTO 220
1080 i=1
1090 l(i)=x(m(i,1))+x(m(i,2))+x(m(i,3))+x(m(i,4))
1100 l=l(i)
1110 IF l<10 THEN 1180
1120 IF l>=11 THEN 1180
1130 IF l>10 THEN 1980
1140 FOR j=1 TO 4
1150 IF x(m(i,j))<>0 THEN 1170
1160 x(m(i,j))=1/8
1170 NEXT j
1180 i=i+1:IF i<=76 THEN 1090
```

```
1190 GOSUB 1400
1200 i=1
1210 IF l(i)=1/2 THEN 2110
1220 IF l(i)=5+3/8 THEN 2110
1230 i=i+1:IF i<=76 THEN 1210
1240 GOSUB 2250
1250 GOTO 810
1260 PRINT" "
1270 PRINT"Do-ya-wan-annuver-go?";
1280 x$=INKEY$:IF x$="" THEN 1280
1290 IF x$="y" OR x$="Y" THEN 130
1300 PRINT:PRINT:PRINT:PRINT"Bye..sucker!!
1310 END
1320 REM
1330 k1=INT((m-1)/16)+1
1340 j2=m-16*(k1-1)
1350 k2=INT((j2-1)/4)+1
1360 k3=m-(k1-1)*16-(k2-1)*4
1370 m=k1*100+k2*10+k3
1380 PRINT m;
1390 RETURN
1400 FOR s=1 TO 76
1410 j1=m(s,1)
1420 j2=m(s,2)
1430 j3=m(s,3)
1440 j4=m(s,4)
1450 l(s)=x(j1)+x(j2)+x(j3)+x(j4)
1460 NEXT s
1470 RETURN
1480 FOR i=1 TO 64
1490 IF x(i)<>0 THEN 1560
1500 x(i)=5
1510 m=i
1520 PRINT"Machines likes";
1530 GOSUB 1330
1540 PRINT" "
1550 GOTO 220
1560 NEXT i
1570 PRINT"The game is a draw."
1580 GOTO 1260
1590 FOR k=1 TO 18
1600 p=0
1610 FOR i=4*k-3 TO 4*k
1620 FOR j=1 TO 4
1630 p=p+x(m(i,j))
1640 NEXT j,i
1650 IF p<4 THEN 1690
1660 IF p<5 THEN 1720
1670 IF p<9 THEN 1720
1680 IF p<10 THEN 1720
1690 NEXT k
1700 GOSUB 2250
1710 GOTO 980
1720 s=1/8
1730 FOR i=4*k-3 TO 4*k
1740 GOTO 2120
1750 NEXT i
1760 s=0
1770 GOTO 1730
1780 DATA 1,49,52,4,13,61,64,16,22,39,23,38,26,42,27,43
1790 DATA 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20
1800 DATA 21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,38
1810 DATA 39,40,41,42,43,44,45,46,47,48,49,50,51,52,53,54,55,56
1820 DATA 57,58,59,60,61,62,63,64
```

```
1830 DATA 1,17,33,49,5,21,37,53,9,25,41,57,13,29,45,61
1840 DATA 2,18,34,50,6,22,38,54,10,26,42,58,14,30,46,62
1850 DATA 3,19,35,51,7,23,39,55,11,27,43,59,15,31,47,63
1860 DATA 4,20,36,52,8,24,40,56,12,28,44,60,16,32,48,64
1870 DATA 1,5,9,13,17,21,25,29,33,37,41,45,49,53,57,61
1880 DATA 2,6,10,14,18,22,26,30,34,38,42,46,50,54,58,62
1890 DATA 3,7,11,15,19,23,27,31,35,39,43,47,51,55,59,63
1900 DATA 4,8,12,16,20,24,28,32,36,40,44,48,52,56,60,64
1910 DATA 1,6,11,16,17,22,27,32,33,38,43,48,49,54,59,64
1920 DATA 13,10,7,4,29,26,23,20,45,42,39,36,61,58,55,52
1930 DATA 1,21,41,61,2,22,42,62,3,23,43,63,4,24,44,64
1940 DATA 49,37,25,13,50,38,26,14,51,39,27,15,52,40,28,16
1950 DATA 1,18,35,52,5,22,39,56,9,26,43,60,13,30,47,64
1960 DATA 49,34,19,4,53,38,23,8,57,42,27,12,61,46,31,16
1970 DATA 1,22,43,64,16,27,38,49,4,23,42,61,13,26,39,52
1980 FOR j=1 TO 4
1990 IF x(m(i,j))<>1/8 THEN 2080
2000 x(m(i,j))=5
2010 IF l(i)<5 THEN 2040
2020 PRINT"Let's see you get out of this:":PRINT"Machine moves to ";
2030 GOTO 2050
2040 PRINT"You lucky devil!":PRINT"Just in the nick of time":PRINT"Mach
 to ";
2050 m=m(i,j)
2060 GOSUB 1330
2070 GOTO 220
2080 NEXT j
2090 PRINT"Machine concedes this game."
2100 GOTO 1260
2110 s=1/8
2120 IF i-INT(i/4)*4>1 THEN 2150
2130 a=1
2140 GOTO 2160
2150 a=2
2160 FOR j=a TO 5-a STEP 5-2*a
2170 IF x(m(i,j))=s THEN 2200
2180 NEXT j
2190 GOTO 1750
2200 x(m(i,j))=5
2210 m=m(i,j)
2220 PRINT"Machine takes";
2230 GOSUB 1330
2240 GOTO 220
2250 FOR i=1 TO 64
2260 IF x(i)<>1/8 THEN 2280
2270 x(i)=0
2280 NEXT i
2290 RETURN
2300 FOR xx=1 TO 9:PRINT:NEXT:FOR i=1 TO 4
2310 FOR j=1 TO 4
2320 FOR i1=1 TO j
2330 PRINT"   ";
2340 NEXT i1
2350 FOR k=1 TO 4
2360 q=16*i+4*j+k-20
2370 IF x(q)<>0 THEN 2390
2380 PRINT"( )      ";
2390 IF x(q)<>5 THEN 2410
2400 PRINT"(M)      ";
2410 IF x(q)<>1 THEN 2430
2420 PRINT"(Y)      ";
2430 IF x(q)<>1/8 THEN 2450
2440 PRINT"( )      ";
2450 NEXT k
2460 PRINT
2470 NEXT j:PRINT
2480 NEXT i
2490 RETURN
2500 PRINT"Boo Boo, retype it--":GOTO 240
2510 END
```

The Forth Programming Language

By Michael Howard

Many of you probably know how to program in BASIC, and although this language is often put down by critics as a lousy programming language it is in fact a very good and complex tool.

All programming languages have their advocates, my pesonal favourites being Pascal, C and this one . . . FORTH. Few devotees are as impassioned (crazy!) as those who support FORTH. FORTH is incredible flexible, fast (typically 20 to 30 times faster than an equivalent BASIC program), and very, very compact (in other words a FORTH program will take up very little memory). So I thought it would be a good idea to develop a way of letting Sega owners get the taste of FORTH without having to buy the language. The program that follows let's you mess about with FORTH, but remember, it is not a full FORTH, and it is very slow, because it is written in BASIC.

The following program is approx. 9k long, so it is a bit of a typing chore, but when you consider that you end up with a new language (for free), the effort is more than worth while, In fact a friend of mine, who I let test the program, (so that I could get constructive criticism and help with the debugging) does nothing all day except play with this FORTH emulator!

The FORTH programming languages was developed by a gentleman by the name of Charles Henry Moore, he found that the languages of the day could not do what he wanted them to do, so he invented his own. The computer he was working on was called an IBM 1130, and in the 1970's these beasts were the "ultimate in computers!". Moore believed he had developed a Fourth Generation Language (BASIC is Third and a half generation!) so wanted to call the language "Fourth," but unfortunately the IBM would only allow program names to be of 5 or less digits in length, so Moore had to change the name to FORTH! This is a true story!!!!

FORTH differs from many other languages, such as BASIC, because it comes with a standard set of words which the programmer can use to develop his or her own commands, which makes FORTH fast, freaky, functional and flexible (what a mouthful!!) and trendy!

However, as the authors of the book "FORTH" (Salmon, Tosserand and Toulout; Macmillan, 1984) point out, this flexibility means that FORTH can in fact be written in FORTH!! This emulator won't let you do that, but it will let you create up to 500 new words, which the system will execute as though they were programming words provided by the raw version of the language.

Reverse Polish Notation . . . RPN

One of the more unusual features of FORTH is it's arithmetic. When you want to add two numbers together, say 42 and 69, in BASIC, you would enter the following:-

PRINT 42 + 69

In FORTH you'd need to enter:-

42 69 +.

You enter the first number, then a space (spaces are really important in FORTH, as you'll see soon), then the second number, then the operation you wish to perform, in this case + , then follow it by a dot, this tells FORTH that you want the result of the calculation to be printed out. If you, omit this dot, FORTH will still work out the value, but it won't print it on the screen. This kind of arithmetic, working from left to right, is called Reverse Polish Notation.

The Stack

The stack is one of the most fundamental principles of FORTH, so it is well worth the effort of trying to understand what it is and how it works (another reason for learning how it works, is because in the future, when you start to learn Machine Code, you'll have to be able to manipulate the stack . . . and what you learn here will be applicable to the Machine code environment).

The stack is like a tall mound of paper. You can only add or remove pieces from the top of the stack of paper, and the last piece of paper you put on the pile, will be the first to be taken off.

When you enter the number 42 followed by a space, a computer running FORTH effectively writes this number onto a piece of paper and places it in the top of the stack of paper. It then writes the next number (69) onto a piece of paper, and puts that on the top of the stack of paper, so now, 69 is on the top of the pile, whilst 42 is second to the top . . . think about it a little.

The computer then comes up to the " + " instruction, which is a word in the computer's 'dictionary' of words, " + " tells the computer to take the top two numbers from the top of the stack, add them together, then place the result onto the top of the stack. This is what **42 69 +** does. Finally, when FORTH bumps into '.', the computer prints the result out. The dot means take the value off the top of the stack and print it out.

As I said earlier, if you don't include the dot command, the computer will still work out the problem, but will leave on the top of the stack.

Making the Stack visible

The stack provided with my program can be up to 20 'sheets' of paper deep. The sheets can only store numbers, and as we saw before, a number is written on the top sheet, then the sheet which used to be at the top is now second, the sheet that was second is now third and so on . . . The program let's you see the top four elements of the stack, called a 'Visible Stack', so you can see what is going on.

When you first run the program, you will be asked if you want to have the stack printed out, if you say YES, then after each round of instructions, the stack will be displayed on the screen, thus:

Stack:
1..--
2..--
3..--
4..--

Ok

The double minus sign '--', means that the stack is empty at the point (or if you like, the sheet of paper is blank). You can prove this by entering a dot (.) after the OK appears. This, you'll recall, is to print out the top element of the stack. When the stack is empty, as in this case, you'll get a **Stack Empty** error. So let's have a bash at working out our little sum, and taking a look at the stack whilst the program is running.

When the Sega comes up with the Ok message, it means that it is ready to accept a command, somewhat similar to READY in BASIC. Type in the following sum, making sure you put spaces between each 'Word':

42 69 +

After a short pause, you'll get the answer back (111). As you can see the command has all been entered on one line, to see what the stack is up to, we'll have to break down the command into sub-commands, ie **42 69 +** . is made up of four instructions, **42, 69 +** , and . so we could do the following instead:-

1) Type in **42** now press the **CR** key.
2) Type in **69** now press the **CR** key
3) Type in + now press the **CR** key.
4) Type in . now press the **CR** key.

You'll notice that at each stage the stack alters.

When you press 42, the stack will look like this:

Stack:
1..42
2..--
3..--
4..--

Ok

You can see 42, at the top of the stack.

When you, enter 69, the stack will look like this:

Stack:
1..69
2..42
3..--
4..--

Ok

Notice how the 42 has been PUSH'ed down and 69 is placed on the top. Now when the + is encountered, a strange thing happens, firstly the top two numbers on the stack are removed and then added together, the result is then put at the top the stack, don't forget that the top two numbers have now been removed. After the + operation the stack will become:

Stack:
1..111
2..--
3..--
4..--

Ok

In other words . . . empty!

Now, just mess about with a few simple calculations and take a look at the stack after each command, but don't forget those spaces!!! and mess with /, +, -, and *.

All FORTH operations boil down to PUSH'ing onto the stack, and POP'ing it off.

Defining Your Own Words

The most exciting thing about FORTH is the ability to create your very own new words, and we'll look at this next.

You'll recall, from the earlier arithmetic examples you've tried that the dot is used to pop the top value off the stack and print it on the screen. There may be times when you want to print out the value but still want the value on the top of the stack to stay there. Dot, as you have seen is destructive-you lose the top number on the stack by popping it off.

The word DUP (for 'DUPlicate') gets around this little problem by duplicating the value at the top of the stack onto the top of the stack, in other words, if the stack looks like this:

Stack:
1..20
2..--
3..--
4..--

Ok

Then after DUP, the stack would become:

Stack:
1..20
2..20
3..--
4..--

Ok

See what has happened, the number 20, has been duplicated, So now you could pop the value of the top of the stack and leave a copy of that number on the top of the stack.

We can invent a word of our own, PRINTOUT, to duplicate the number on the top of the stack and print it out. You create words, by using the following format:-

: Program Body ;

Note that a ':' starts off the definition of a new word, and a ';' ends the definition.

So our word PRINTOUT, would be:

: PRINTOUT DUP . ;

(note the spaces)

In this example, we start off with a colon, follow it with a space, then the program body (DUP .) followed by a space and then semicolon.

From this point onwards you, can simply use PRINTOUT in your FORTH Programs, and your computer will automatically perform the **DUP .** for you. The program, as it stands, allows you to create up to 500 new words.

Here is another word:

: SQUARED DUP * . ;

In this example, whenever we call SQUARED, the number on the top of the stack is duplicated, the top two numbers (the same number) are multiplied together and the result is printed out. Here's an example: If we said 4 SQUARED (don't forget, the 4 has to go on the stack before SQUARED is used) we would get an answer of 16. Here is how it works:

1) 4 is put on the stack. The stack now looks like this:

Stack:
1..4
2..--
3..--
4..--

---

2) SQUARED is now invoked.
3) The first instruction in SQUARED is DUP, so the stack is now:

Stack:
1..4
2..4
3..--
4..--

---

4) The next instruction in SQUARED it is *, so the top two values on the stack are removed and multiplied together, the answer is then put on the stack. The stack is now:

Stack:
1..16
2..--
3..--
4..--

---

5) The next part of SQUARED instruction is . which prints out what is on the top of the stack, in this case 16 is on the top, so 16 is printed out, and the stack is left bare.

Imagine if we wanted the value of 4 SQUARED (16) to be left at the top of the stack, we could write this:

: SQUARED DUP * DUP . ;

The 'DUP *' bit does the usual bit, the 'DUP .' bit makes sure that the value at the top of the stack is printed out, but only after a copy is made.

Now this is where the beauty of FORTH comes into it's own, if we'd already defined PRINTOUT as **DUP .** then we could alter out new SQUARED program to:

: SQUARED DUP * PRINTOUT ;

See what is going on, PRINTOUT has replaced DUP . good stuff eh?!?!

FORTH always uses the most recently defined version of a word, searching it's dictionary from the newest word to the oldest, so you can safely redefine a word and know that it will use the latest version.

You can see a hint of real magic of FORTH in this inclusion of PRINTOUT within SQUARED, you could define lots of words and include them in newer words as oftens as you like!! This is the sheer power of FORTH!

To forget a word from the dictionary, type in FORGET followed by the word to forget, but be warned that when you, do this FORTH will also forget all the newer words defined after the word you told it to FORGET!

Forth in action

Forth works in more or less in the following way:

— The user types in something into the computer after the OK prompt.

— FORTH looks through it's dictionary of words to see if any of them match the word which has been entered.

— If it finds a match, either within the dictionary provided with the program, or within the dictionary of words the user has provide, it executes the word then goes to the next element in the input.

— If it doesn't find a match, it assumes the word is a number.

— If it isn't a number, then you made a mistake.

You'll find that this program is well furnished with error messages, which (along with the visible stack) should make it pretty easy to keep aware of what is going on in the program.

NOTE: Very few, if any, FORTH compilers give a visible stack.

The usefulness of being able to define words is illustrated by the following examples. Suppose you want a word called CIRCLE, which would take a number off the top of the stack, and treat it as the diameter of the circle, you'd only need to enter something like 23 CIRCLE to get your Sega to print out the circumference of a circle with a diameter of 23 units.

The word CIRCLE could be defined as follows (where 355/113 is an integer approximation of PI, since most FORTHs work in integers, rather than floating point numbers, such as 1,1.3.1415926536 etc):

: CIRCLE 255 113 * / . .'
IS THE CIRCUMFERENCE' ;

This works by multiplying the value on the top of the stack by 355, this new value is then divided by 113, this result is then printed out, followed by the message IS THE CIRCUMFERENCE.

The new words in this are:-

* multiply
/ divide
.'print out a string of letters, Eg;.'HELLO' would print out HELLO, note the spaces.

If you made the above word, then all that you would have to do is enter a number (or let the computer take a number off the top of the stack) for the diameter and the computer will do the rest. You'll find that you'll have a lot of fun defining all your own words and using them in your programs.

The built in Vocabulary

There are two widely used 'standard' versions of FORTH, fig-FORTH (From the Forth Interest Group) and FORTH-79. The program that I have provided sticks mostly to the Forth-79 standard, but there are a few Fig-Forth equivalents, such as Fig's MINUS and 79's NEGATE are both supported, and they are the same.

The vocabulary supplied with the program is quiet extensive, but by no means complete. I will now explain each word provided.

+,-,/ and * these all take two numbers off the top of the stack and operate on them, eg add, subtract etc and then push the result onto the stack.

/ and MOD these are the same in that they divide, but / puts the quotient on the top of the stack, and MOD puts the remainder on the stack. As an example 11/5, the quotient is 2 (5 goes into 11 twice) and the remainder is 1. So 11 5 / would leave 2 on the stack 11 5 MOD would leave 1 on the stack.

/MOD is one word and puts the quotient and remainder on the stack, with the quotient at the top.

*/ is, once again, one word. It multiplies, then divides leaving the result on the top of the stack. Note that this requires 3 numbers on the stack before the command is used.

***/MOD** is similar to */, returning the quotient and remainder, with the quotient on the top.

** takes two numbers from the top of the stack and raises the second number to the power of the first one on the stack, returning the result on the stack.

NEGATE and MINUS take the top number off the stack, and make it negative, the value is then put back on the stack.

ABS is the same as BASIC's ABS, it takes the value off the top of the stack, take sthe absolute value (eg 10 becomes 10, -11 becomes 11, -42 becomes 42 etc.) and places it on the stack.

MAX and MIN compare the top two numbers on the stack, leaving only the largest (MAX) or the smallest (MIN) behind.

DUP we have already met this, it duplicates the top of the stack, so where there is one value on the top of the stack, you get two.

?DUP is similar to DUP, but the duplication is made only when the top of the stack is not zero.

OVER takes the second element on the stack, and copies it over to the top of the stack. So if the stack looks like this:-

Stack:
1..16
2..88
3..--
4..--

Then after OVER, the stack will become:-

Stack:
1..88
2..16
3..88
4..--

See how 88 has been copies to the top of the stack thus pushing 16 and 88 down.

PICK must be preceded by a number. This command selects the numbered element on the stack and copies it to be top of the stack. So if you type 3 PICK, the third element will be copied to the top of the stack.

DROP deletes the top element from the stack, and moves the rest of the stack up.

SWAP causes the top two numbers on the stack to change places.

ROT brings the third element on the stack to the top, whilst the top two are moved down a peg. Eg;

Stack:
1..16
2..55
3..98
4..--

Becomes, after ROT:

Stack:
1..98
2..16
3..55
4..--

ROLL like PICK, must be preceded by a number. It brings this numbered element to the top of the stack, deleting it from it's original position. All the other elements move down a peg.

VLIST lists all the words that the computer knows.

FORGET must be followed by a name, every word defined after this word is deleted from the dictionary. This should be used only with caution.

KEY works just like INKEY$ in BASIC. It waits until you press a key, and that key value is stored on the top of the stack. So if you press the space bar, 32 will be stored on the stack. Check your ASCII table of characters in the users manual.

. (dot) this prints out, as a number, the top value on the stack.

.' (dot quote) this must be followed by a space, and then followed by some text. Eg .' HELLO ESTER ' will print out HELLO ESTER.
' (tick) prints out the description of a word, so ' PRINTOUT will give you all the commands that make up the word PRINTOUT.

EMIT prints out the ASCII character of a number, so 42 EMIT will print out an asterisk.

SPACES can be used to print out spaces, so 10 SPACES will print out 10 spaces!

CR will move the print output to the next line down.

DO . . . LOOP is similar to FOR . . . NEXT in BASIC. It repeats everything between DO and LOOP. The number of repetitions is the difference between the top two numbers on the stack. If 11 were on the top of the stack, and 1 the next number down, then the loop would run for 10 times.

You're bound to have met **IF . . . THEN** from BASIC, and even though they do in fact perform the same function, the FORTH version may appear to be a little wierd, to say the least!

The condition being tested comes before the IF, and if the condition is true then all the commands between the IF and THEN are executed. If it is false then the commands after the THEN are executed. IF/THEN in FORTH can evaluate =, > </< > as well as check if a number at the top of the stack is equal to, not equal to, greater than or less than zero.

The word **NOT** causes the IF/THEN to look in the opposite direction.

Finally, in our standard vocab there are some special arithmetic words to add one to the number on the top of the stack (**1+**), or subtract one (**1-**), the other commands are **2-**, **2+**, **2/** and **2***.

I have also put some non-standard words in, like **STACK** to turn the viewing stack on or off. If the stack is on then after a STACK word, then stack will be turned off and vice versa.

ABORT quits an action.

LLIST will list all the words in the memory and their descriptions.

To help clear up a few of the mysteries, here are some programs.

The first of which is a game of "Guess my Number". The computer thinks up a number between 10 and 100, and you are given 10 chances at guessing the number.

The game itself, is held in the word GAME, this in turn uses other words, which in turn uses other words etc.

The program may look quite complex, but if you take it to bits and play about with FORTH for a while it'll all click.

To run the game just enter GAME, but only after you have entered the program words.

```
: GAME THINKNUM 11 1 DO GETGUESS TEST
LOOP END ;
: THINKNUM 90 10 RAND ;
: GETGUESS CR . ' Enter guess ' KEY 48 — KEY 48
— . ' Ta! ' SWAP 10 * + ;
: TEST SAME? BIGGER? SMALLER? DROP ;
: END . ' Sorry. Time is up! ' CR . ' I was thining of ' .;
: SPINNER DUP ROT DUP ROT ;
: SAME? SPINNER = IF.'You did it!' ABORT THEN.
;
: BIGGER? SPINNER > IF . ' Too big ' THEN ;
: SMALLER? SPINNER SWAP < IF . ' Too small '
THEN SWAP;
```

This next program is simple:-

```
: STAR 42 EMIT ;
: STARS 11 1 STAR 11 1 DO STAR LOOP;
```

Now whenever you enter STARS, you will get a row of 10 stars.

Try experimenting. I will love to hear from anyone who really takes to FORTH and decides to write a few simple programs, I will be more than happy to publish them in the magazine, and perhaps even set up a little FORTH interest group, but that of course depends on you!

I hope you like it.

```
100 REM    ********************************
110 REM    *      A Forth Emulator        *
120 REM    * Written By Michael Howard *
130 REM    *      9th March 1986          *
140 REM    ********************************
150 REM
160 REM
170 REM       ------------------------
180 REM       ! Make sure that the  !
190 REM       !   program is using  !
200 REM       !   CAPITAL letters   !
210 REM       ------------------------
220 REM
230 REM
240 REM
250 c$="":e$="":nc=0
260 DIM a(21),b(21),c(21),n$(500),m$(500)
270 FOR j=1 TO 21:a(j)=1E-10:b(j)=a(j):NEXT j
280 CLS
290 IF INKEY$<>"" THEN 290
300 GOTO 640
310 REM Print out Stack
320 REM Print out the Stack
```

```
330 IF sflag=0 THEN RETURN
340 PRINT TAB(5);c$
350 PRINT TAB(17);e$
360 PRINT:PRINT TAB(5);"Stack:"
370 FOR q=1 TO 4
380 IF a(q)<>1E-10 THEN PRINT TAB(8);q;"..";a(q)
390 IF a(q)=1E-10 THEN PRINT TAB(8);q;"..--"
400 NEXT q
410 PRINT TAB(3);"----------------------":PRINT
420 RETURN
430 REM Pop the Stack
440 REM
450 e=a(1)
460 FOR j=2 TO 21:a(j-1)=a(j):NEXT j
470 a(21)=1E-10
480 RETURN
490 REM Push the Stack
500 REM
510 FOR j=1 TO 20:b(j+1)=a(j):NEXT j
520 FOR j=2 TO 21:a(j)=b(j):NEXT j
530 a(1)=e
540 a(21)=1E-10
550 RETURN
560 REM Remove Untwanted Spaces
570 q=LEN(c$)
580 j=0
590 j=j+1
600 IF MID$(c$,j,1)<>" " THEN 620
610 IF MID$(c$,j+1,1)=" " THEN c$=LEFT$(c$,j)+MID$(c$,j+2):q=q-1:GOTO 610
620 IF j<q THEN 590
630 RETURN
640 PRINT:PRINT:PRINT"Do you want the stack":PRINT"to be printed (Y/N)?"
650 w$=INKEY$:IF w$="" THEN 650
660 sflag=0:IF w$="y" OR w$="Y" THEN sflag=1
670 CLS:GOSUB 320
680 REM Input bit
690 REM
700 GOSUB 320:df=0:PRINT"Ok":INPUT c$
710 IF c$="" THEN END
720 c$=c$+" "
730 GOSUB 560
740 REM Scan Input
750 j=0
760 j=j+1
770 IF LEN(c$)=0 AND df=1 THEN RETURN
780 IF LEN(c$)=0 THEN 700
790 IF MID$(c$,j,1)<>" " THEN 760
800 e$=LEFT$(c$,j-1)
810 c$=MID$(c$,j+1)
820 REM Perform Deed
830 IF e$="ABORT" THEN 690
840 IF e$="CR" THEN PRINT:GOTO 750
850 IF e$="LLIST" THEN FOR u=nc TO 1 STEP -1:lprint n$(u),m$(u):NEXT:GOTO 750
860 IF e$="STACK" THEN sflag=-sflag+1:GOTO 750
870 REM Some Forth Words
880 kw=0
890 IF e$=":" THEN kw=-99:GOSUB 1350
900 IF e$="+" THEN kw=1
910 IF e$="-" AND MID$(c$,3,2)<> "IF" THEN kw=2
920 IF e$="*" THEN kw=3
930 IF e$="/" THEN kw=4
940 IF e$="MOD" THEN kw=5
950 IF e$="/MOD" THEN kw=6
960 IF e$="**" THEN kw=7
970 IF e$="SWAP" THEN kw=8
980 IF e$="*/" THEN kw=9
990 IF e$="*/MOD" THEN kw=10
1000 IF e$="NEGATE" OR e$="MINUS" THEN kw=11
1010 IF e$="ABS" THEN kw=12
1020 IF e$="MAX" THEN kw=13
1030 IF e$="MIN" THEN kw=14
```

```
1040 IF e$="DUP" OR e$="?DUP" THEN kw=15
1050 IF e$="OVER" THEN kw=16
1060 IF e$="PICK" THEN kw=17
1070 IF e$="DROP" THEN kw=18
1080 IF e$="ROT" THEN kw=19
1090 IF e$="ROLL" THEN kw=20
1100 IF e$=".'" THEN kw=21
1110 IF e$="." THEN kw=22
1120 IF e$="EMIT" THEN kw=23
1130 IF e$="VLIST" THEN kw=24
1140 IF e$="FORGET" THEN kw=25
1150 IF LEFT$(e$,1)="'" THEN kw=26
1160 IF e$="KEY" THEN kw=27
1170 IF e$="RAND" THEN kw=28
1180 IF e$="DO" THEN kw=29
1190 IF e$="SPACES" THEN kw=30
1200 IF e$="1+" OR e$="1-" OR e$="2+" OR e$="2-" THEN kw=32
1210 IF e$="2*" OR e$="2/" THEN kw=32
1220 IF e$="=" OR e$="<" OR e$=">" OR e$="0=" OR e$="0<" OR e$="0>" THEN kw=31
1230 IF (e$="-" AND kw=0) OR e$="NOT" THEN kw=31
1240 IF kw>0 AND kw<8 THEN GOSUB 1610:GOTO 750
1250 IF kw>8 AND kw<17 THEN ON kw-8 GOSUB 1910,1960,2030,2060,1610,1610,2090,213
0
1260 IF kw>16 AND kw<23 THEN ON kw-16 GOSUB 2180,2250,2280,2180,2330,2430
1270 IF kw>22 AND kw<29 THEN ON kw-22 GOSUB 2480,2510,2640,2800,2870,2910
1280 IF kw>28 THEN ON kw-28 GOSUB 2950,3100,3130,3450
1290 REM Action of Pushing
1300 cf=0:IF kw=0 AND LEN(e$)>0 THEN cf=1
1310 IF cf=1 THEN IF ASC(e$)>44 AND ASC(e$)<58 THEN e=VAL(e$):GOSUB 490
1320 IF cf=1 THEN IF ASC(e$)>57 THEN GOSUB 1520:GOTO 720
1330 GOTO 750
1340 REM Define your new words!
1350 REM
1360 IF nc<500 THEN nc=nc+1
1370 x=0
1380 x=x+1
1390 IF MID$(c$,x,1)=";" THEN r$=LEFT$(c$,x-1):c$=MID$(c$,x+1):GOTO 1420
1400 IF x<LEN(c$) THEN 1380
1410 PRINT TAB(25);" Missing ';' in definition":c$="":RETURN
1420 REM
1430 j=0
1440 j=j+1
1450 IF MID$(r$,j,1)=" " THEN 1480
1460 IF j<LEN(r$) THEN 1440
1470 PRINT TAB(25);"Error in input":c$="":RETURN
1480 n$(nc)=LEFT$(r$,j-1)
1490 m$(nc)=MID$(r$,j+1)
1500 RETURN
1510 REM
1520 REM
1530 j=nc+1
1540 j=j-1
1550 IF e$=n$(j) THEN 1580
1560 IF j>0 THEN 1540
1570 PRINT TAB(25);"Word not defined":kw=-99:RETURN
1580 c$=m$(j)+c$
1590 RETURN
1600 REM Two Number Operations
1610 t1=0:t2=0:t4=0:k$=""
1620 GOSUB 440:t1=e
1630 GOSUB 440:t2=e
1640 IF e$="MAX" OR e$="MIN" THEN 1860
1650 IF e$="+" THEN t3=t2+t1
1660 IF e$="-" THEN t3=t2-t1
1670 IF e$="*" THEN t3=t2*t1
1680 IF (e$<>"/" AND e$<>"/MOD" AND e$<>"MOD") THEN 1700
1690 IF t1=0 THEN PRINT TAB(25);"Div by 0":c$="":RETURN
1700 IF e$="/" OR e$="/MOD" THEN t3=INT(t2/t1)
1710 IF e$="MOD" THEN t3=t1*(t2/t1-INT(t2/t1))
1720 IF e$="**" THEN t3=t2^t1
1730 IF e$="/MOD" THEN 1770
```

```
1740 IF e$="SWAP" THEN 1820
1750 e=INT(t3+0.5):GOSUB 500:RETURN
1760 REM
1770 REM /MOD
1780 t4=t1*(t2/t1-INT(t2/t1))
1790 e=INT(t4+0.5):GOSUB 500
1800 e=INT(t3+0.5):GOSUB 500:RETURN
1810 REM
1820 REM SWAP
1830 e=t1:GOSUB 500
1840 e=t2:GOSUB 500:RETURN
1850 REM
1860 REM MAX and MIN
1870 a=t1:b=t2:IF t1>t2 THEN a=t2:b=t1
1880 e=a:IF e$="MAX" THEN e=b
1890 GOSUB 500:RETURN
1900 REM
1910 REM */
1920 GOSUB 440:p3=e:GOSUB 440:p2=e:GOSUB 440:p1=e
1930 e=INT(p1*p2/p3):GOSUB 500
1940 RETURN
1950 REM
1960 REM */MOD
1970 GOSUB 440:cq=e:GOSUB 440:bq=e:GOSUB 440:aq=e
1980 dq=aq*bq/cq
1990 e=INT(cq*((dq)-INT(dq))):GOSUB 500
2000 e=INT(dq):GOSUB 500
2010 RETURN
2020 REM
2030 REM Negate/Minus
2040 GOSUB 440:e=-e:GOSUB 500:RETURN
2050 REM
2060 REM ABS
2070 GOSUB 440:e=ABS(e):GOSUB 500:RETURN
2080 REM
2090 REM DUP  ?DUP/-DUP
2100 GOSUB 440:P1=E:IF e=0 AND (e$="?DUP" OR e$="-DUP") THEN GOSUB 500:RETURN
2110 e=P1:GOSUB 500:GOSUB 500:RETURN
2120 REM
2130 REM OVER
2140 GOSUB 440:xw=e:GOSUB 440:yw=e
2150 e=yw:GOSUB 500:e=xw:GOSUB 500:e=yw:GOSUB 500
2160 RETURN
2170 REM
2180 REM PICK ROLL
2190 GOSUB 440:fr=e-1
2200 FOR t=1 TO fr:GOSUB 440:c(t)=e:NEXT t
2210 GOSUB 440:xr=e:IF kw=17 THEN GOSUB 500
2220 FOR t=fr TO 1 STEP-1:e=c(t):GOSUB 500:NEXT t
2230 e=xr:GOSUB 500:RETURN
2240 REM
2250 REM DROP
2260 GOSUB 440:RETURN
2270 REM
2280 REM ROT
2290 GOSUB 440:g3=e:GOSUB 440:g2=e:GOSUB 440:g1=e
2300 e=g2:GOSUB 500:e=g3:GOSUB 500:e=g1:GOSUB 500
2310 RETURN
2320 REM
2330 REM .'
2340 x=0
2350 x=x+1
2360 IF MID$(c$,x,1)="'" THEN 2390
2370 IF x<LEN(c$) THEN 2350
2380 PRINT TAB(25);"No closing ' in your input":RETURN
2390 PRINT LEFT$(c$,x-1);
2400 c$=MID$(c$,x+1)
2410 RETURN
2420 REM
2430 REM .
2440 GOSUB 440
```

```
2450 IF e<>1E-10 THEN PRINT e;" ";:RETURN
2460 PRINT"Stack empty":PRINT:e=1E-10:GOSUB 500:RETURN
2470 REM
2480 REM EMIT
2490 GOSUB 440:PRINT CHR$(e);:RETURN
2500 REM
2510 REM VLIST
2520 PRINT:IF nc=0 THEN 2550
2530 FOR j=nc TO 1 STEP -1:IF n$(j)<>"" THEN PRINT n$(j),
2540 NEXT j
2550 PRINT "+","-","*","/","MOD","/MOD","**","SWAP","*/","*/MOD",
2560 PRINT"NEGATE","MINUS","ABS","MAX","MIN","OVER","PICK","DROP",
2570 PRINT"ROT","ROLL",".'","'","-",":",";","EMIT","VLIST","FORGET",
2580 PRINT"KEY","DO","LOOP","SPACES","IF","THEN",
2590 PRINT"DUP","?DUP","-DUP","=","<",">","0=","0<","0>",
2600 PRINT"NOT","1+","1-","2+","2-","2*","2/"
2610 PRINT"This system also supports..":PRINT"STACK ABORT LLIST RAND CR"
2620 PRINT:RETURN
2630 REM
2640 REM FORGET
2650 j=0
2660 j=j+1
2670 IF j>LEN(c$) THEN PRINT TAB(25);"Error in FORGET":kw=-99:RETURN
2680 IF MID$(c$,j,1)=" " THEN 2700
2690 GOTO 2660
2700 f$=LEFT$(c$,j-1):c$=MID$(c$,j+1):flag=0
2710 FOR j=nc+1 TO 1 STEP -1
2720 IF n$(j)=f$ THEN n$(j)="":m$(j)="":flag=1:drop=j
2730 NEXT j
2740 IF flag=0 THEN PRINT TAB(25);"Word ";f$;" not in dictionary.":c$=MID$(c$,LE
N(f$),1):kw=-99:RETURN
2750 zn=nc:FOR j=nc TO drop STEP-1
2760 n$(j)="":m$(J)="":zn=zn-1
2770 NEXT j:nc=zn
2780 RETURN
2790 REM
2800 REM '
2810 PRINT:e$=MID$(e$,2):f1=0
2820 FOR j=1 TO nc:IF n$(j)=e$ THEN f1=j
2830 NEXT j:IF f1=0 THEN PRINT TAB(25);"Error - ";e$;" not in dictionary.":RETUR
N
2840 PRINT TAB(10);e$;" Ok":PRINT:PRINT TAB(12);m$(f1)
2850 RETURN
2860 REM
2870 REM KEY
2880 w$=INKEY$:IF w$="" THEN 2880
2890 e=ASC(w$):GOSUB 500:RETURN
2900 REM
2910 REM RAND (NOT a Standard Forth word)
2920 GOSUB 440:y1=e:GOSUB 440:y2=e
2930 e=INT(RND(8)*y2+y1):GOSUB 500:RETURN
2940 REM
2950 REM DO loops
2960 jh=0
2970 jh=jh+1
2980 IF MID$(c$,jh,1)=" " THEN IF MID$(c$,jh+1,4)="LOOP" THEN 3010
2990 IF jh<LEN(c$) THEN 2970
3000 PRINT TAB(25);"Error in DO..LOOP":RETURN
3010 df=1
3020 o$=LEFT$(c$,jh):i$=MID$(c$,jh+6)
3030 GOSUB 440:12=e:GOSUB 440:11=e
3040 st=1:IF 12>=11 THEN st=-1:11=11+2
3050 FOR o=12 TO 11-1 STEP st
3060 c$=o$:GOSUB 720
3070 NEXT o:REM this is a letter o,not a number 0
3080 c$=i$:df=0:RETURN
3090 REM
3100 REM SPACES
3110 GOSUB 440:FOR j=1 TO e:PRINT" ";:NEXT j:RETURN
3120 REM
3130 REM IF...THEN
```

```
3140 IF LEFT$(c$,2)="IF" THEN C$=MID$(c$,4)
3150 IF LEFT$(c$,6)=" NOT IF" THEN c$=MID$(c$,8):GOSUB 3350
3160 bj=0
3170 bj=bj+1
3180 IF MID$(c$,bj,4)="THEN" THEN 3210
3190 IF bj<LEN(c$) THEN 3170
3200 PRINT TAB(25);"Error in IF..THEN":RETURN
3210 u$=LEFT$(c$,bj-1):c$=MID$(c$,bj+5)
3220 GOSUB 440:k2=e
3230 IF LEFT$(e$,1)<>"0" THEN GOSUB 440:k1=e
3240 true=0
3250 IF e$="=" AND k2=k1 THEN true=1
3260 IF (e$="<>" OR e$="-") AND k2<>k1 THEN true=1
3270 IF e$="<" AND k1<k2 THEN true=1
3280 IF e$=">" AND k1>k2 THEN true=1
3290 IF e$="0="AND k2=0 THEN true=1
3300 IF e$="0<>" AND k2<>0 THEN true=1
3310 IF e$="0<" AND k2<0 THEN true=1
3320 IF e$="0>" AND k2>0 THEN true=1
3330 IF true=0 THEN RETURN
3340 c$=u$+c$:RETURN
3350 REM
3360 IF e$="=" THEN e$="<>":RETURN
3370 IF e$="-" THEN e$="=":RETURN
3380 IF e$="<" THEN e$=">":RETURN
3390 IF e$=">" THEN e$="<":RETURN
3400 IF e$="0=" THEN e$="0<>":RETURN
3410 IF e$="0<" THEN e$=0>":return
3420 IF e$="0>" THEN e$="0<":RETURN
3430 PRINT TAB(25);"Error in NOT":RETURN
3440 REM
3450 REM 1+ 1- 2+ 2- 2* 2/
3460 GOSUB 440
3470 IF e$="1+" THEN e=e+1
3480 IF e$="1-" THEN e=e-1
3490 IF e$="2+" THEN e=e+2
3500 IF e$="2-" THEN e=e-2
3510 IF e$="2*" THEN e=e*2
3520 IF e$="2/" THEN e=e/2:e=INT(e)
3530 GOSUB 500:RETURN
```

How to check for a bias dice

If any of you out there are into playing role playing games such as **Dungeons and Dragons** then you will know just how often a die seems to yield a high number when you need one, and a low number when you need one. This is usually in the mind of the player, but sometimes a die can in fact be bias towards one number, and the way to check for it is by using a standard test called **The Chi-square test**. Now this test is really 'orrible and mathematical, and I won't go into too much depth about it, so all I'll say is that Chi-square is a way of testing any statistical date for randomness.

How to do it:-

Enter the program listed below. The program, when run, will ask you to enter the number of faces on the die, this will usually be 6 sides, unless you are Role Player in which case the number of sides could be 4, 6, 8, 10, 12 or 20.

The computer then asks you for a value of E, this is the expected number of times a face will occur, so if you enter say 10, and the dice is 6 sided then after 60 rolls, each face should appear 10 times, if the dice is completely unbiased. Note that if you enter a number less than 10 for the value of E, then the computer automatically sets the value of E to 10.

The program then asks you if you want to categorise the groups, in general answer with a 0, at this stage.

The next step is entry of data. Roll the die and enter the results until the computer tell's you to stop. If you enter an invalid number, such as a 7 when you are using a 6-sided dice, the computer tells you that you've made a mistake and asks you to enter that value again.

Finally, the value of the Chi-square will be given along with a message indicating the level of bias.

```
10 REM MH
20 REM ****************************************
30 REM *                                      *
40 REM * Note that this program can be        *
50 REM * used with any statistical data       *
60 REM *   to check for randomness            *
70 REM *                                      *
80 REM ****************************************
90 REM
100 DIM ct(20),gp(20),c(20),c1(20)
110 FOR i= 1 TO 20:ct(i)=0:NEXT i
120 FOR i=2 TO 20:READ c(i):NEXT i
130 FOR i=2 TO 20:READ c1(i):NEXT i
140 DATA 2.706,4.605,6.251,7.779,9.236,10.645,12.017,13.362,14.684,15.987
150 DATA 17.275,18.549,19.812,21.064,22.307,23.542,24.769,25.989,27.204
160 DATA 6.635,9.210,11.341,13.277,15.086,16.812,18.475,20.090,21.666,23.209
170 DATA 24.725,26.217,27.688,29.141,30.578,32.000,33.409,34.805,36.191
180 IF c1(20)=36.191 THEN 200
190 PRINT"Error in DATA lines":beep 1:END
200 CLS:PRINT"Chi-square test"
210 INPUT"Enter the no. of faces on die ";nf
220 INPUT"Enter expected number of rolls per face ";np
230 IF np<5 THEN PRINT"This is too low, 10 assumed":np=10
240 n=np*nf
250 PRINT"If you want the categories grouped then enter 1, else enter 0"
260 INPUT gf
270 IF gf=0 THEN 360
280 IF NOT(gf=1) THEN 250
290 INPUT"Enter the number of groups ";ng
300 IF ng<2 OR ng>nf-1 THEN PRINT"Number of groups must be between 2 and";nf-1:G
OTO 290
310 PRINT"Now enter the group code, from 1 to";ng;" for each face."
320 FOR i=1 TO nf
330 PRINT i;:INPUT t
340 IF t<1 OR t>ng THEN PRINT"Error- re-try code":GOTO 330
350 gp(i)=t:NEXT i
360 PRINT"Roll die (";n;" times),"
370 PRINT"Enteriing each result as it is rolled."
380 FOR i=1 TO n
390 PRINT i;:INPUT t
400 IF t<1 OR t>nf THEN PRINT"Invalid entry--try again.":GOTO 390
410 ct(t)=ct(t)+1:NEXT i
420 PRINT"Done. Results are:":PRINT:PRINT
430 FOR i=1 TO nf:PRINT i,ct(i):NEXT i
440 PRINT"Hit a key to continue"
450 IF INKEY$="" THEN 450
460 ch=0:IF gf=1 THEN 510
470 FOR i=1 TO nf
480 ch=ch+(ct(i)-np)*(ct(i)-np)
490 NEXT i
500 ch=ch/np:GOTO 610
510 ch=0:df=0
520 FOR j=1 TO ng:t=0:nc=0
530 FOR i=1 TO nf
540 IF NOT(gp(i)=j) THEN 560
550 t=t+ct(i):nc=nc+1
560 NEXT i:IF nc=0 THEN PRINT"Warning -- group;j;" unassigned.":goto 650
570 df=df+1
580 ch=ch+(t-nc*np)*(t-nc*np)/(nc*np)
590 PRINT"Group";j:" count=";t
600 NEXT j:PRINT"Chi Square=";ch
610 nt=nc:IF gf<>0 THEN nt=df
620 IF ch>c(t) THEN 640
630 PRINT"No evidence of bias":END
640 IF ch>c1(t) THEN 660
650 PRINT"Weak evidence of bias":END
660 PRINT"Bias almost certain.":END
```

GRANDSTAND COMPUTERS LTD

Updated and Reduced Prices

SOFTWARE

If you are having any problems obtaining any of the pieces of software please contact Grandstand Computers enclosing a cheque or postal order.

NOTE SOME NEW STOCKS

CARTRIDGES	PRICE
☐ BASIC LEVEL IIIB	$85.50
☐ MUSIC	$50.00
☐ N/SUB	$25.00
☐ BORDER LINE	$25.00
☐ CONGO BONGO	$25.00
☐ YAMATO	$25.00
☐ STAR JACKER	$25.00
☐ CHAMPION TENNIS	$25.00
☐ MONACO GP	$25.00
☐ CHAMPION BASEBALL	$25.00
☐ SINBAD MYSTERY	$25.00
☐ VIDEO FLIPPER	$25.00
☐ PACAR	$25.00
☐ POP FLAMER	$25.00
☐ ORGUS	$25.00

HOME/PERSONAL/FINANCE CASSETTES	PRICE	
☐ FILE SYSTEM	$13.00	
☐ CHEQUE BOOK RECONCILIATION	$13.00	
☐ LOAN & MORTGAGE CALCULATION	$13.00	
☐ GRAPH AND CHART	$13.30	
☐ HUCAL	$95.00	disc only spread sheet
☐ SEGA BASE	$95.00	disc only database
☐ LOGO	$72.22	disc only language

Books	(Book and Cassette)
TEACH YOURSELF BASIC GAMES PROGRAMMING	$29.00
MORE THAN 50 PROGRAMS FOR THE SEGA SC-3000	$16.75
SEGA BEGINNERS GUIDE	$10.00
SELECTED BACK ISSUES OF SEGA MAGAZINES	$5.20

GAME CASSETTES	PRICE
☐ TOWERS OF HANOI	$13.30
☐ HANGMAN	$13.30
☐ BUGALOO	$13.30
☐ LASER BLAST	$13.30
☐ MUNCHMAN	$13.30
☐ ENTERPRISE ESCAPE	$13.30
☐ MARSMOBILE	$13.30
☐ MARS ADVENTURER	$13.30
☐ KALAH	$13.30
☐ CHESS	$13.30
☐ BIT BYTE	$13.30
☐ CODE BREAKER	$13.30
☐ BACKGAMMON	$13.30
☐ LUMBER JACK	$13.30
☐ TRAMPMAN	$13.30
☐ HELP!	$13.30
☐ THE HOUSE!	$13.30

EDUCATIONAL CASSETTES	PRICE
☐ ROCKET MATHS	$13.30
☐ LEARNING ALPHABET	$13.30
☐ WATCH ME DRAW	$13.30
☐ SPELLING	$13.30
☐ ADDITION	$13.30
☐ SUBTRACTION	$13.30
☐ MULTIPLICATION	$13.30
☐ SPRITE EDITOR	$13.30
☐ TYPING TUTOR	$13.30
☐ LEARNING TO COUNT	$13.30
☐ SHAPE & COLOUR QUIZ	$13.30
☐ SEGA GRAPHIC DESIGNER	$13.30
☐ GEOGRAPHY QUIZ I	$13.30
☐ MATHS HANG UP	$13.30

I enclose payment

☐ Cash ☐ Cheque ☐ Postal order ☐ Visa ☐ Bankcard

NAME . . . . . CARD NUMBER

ADDRESS . . . . . |_|_|_|_|_|_| |_|_|_|_|_|_| |_|_|_|_|_|_|

. . . . . CARD EXPIRY DATE . . . . . ✂

SIGNATURE . . . CHEQUE NO . . . . . .

GRANDSTAND COMPUTERS LTD P.O. BOX 2353 AUCKLAND

How to Make Limitless Mazes

. . . Michael Howard

Many excellent games are marred by a limited capacity for storing mazes and the computing time required to set up the data at the start of each run of the program. When details of each game are stored away as data, a lot of memory is taken up, and this memory could be used to make the game a lot better.

There are ways, however, to overcome this problem so that even a humble 16k Sega can guide the games player round labyrinths which are large enough to tax the most hardened games player.

THE TRICK

The trick to getting a very large maze into a small computer is to store the details of the maze in a mathematical equation. Suitable functions already exist to create random numbers, in fact the random number generator in your Sega is nothing more than an equation! To generate a random maze, that is infinite in EVERY director, and is totally mappable can take up less than 1.5k of memory by using a special function!

Mazes are two or in this case three dimensional structures which can be defined by random functions of two or three co-ordinates. The function indicates whether any given location is open corridor or solid wall. If the location is defined by co-ordinates X,Y and Z, then these values can be passed onto a special function and that function dictates whether walls or corridors exist at those co-ordinates.

The function I chose to dictate the maze is the following:

9000 U = 100 * SQR(X * X + Y * Y):W = U − INT(U)

In this program, a special variable called T is allocated, and this dictates how the maze looks, by altering this value, the maze can be totally re-arranged in an instant. In the program below, you can scatter objects around by saying "If the player is at point X, Y and Z then he/she has found a pot of gold, where X,Y and Z are some numbers determined at the start of the game.

Okay, onto the game. At the start of the game T is set a value of 0.3, this sets the maze to a certain shape, if you alter T to some other value then the maze will be different. Next you are told what is around you, and asked for a command. At first you are on the surface, you can only go down into the maze if you meet a shaft, when you go down you are in the maze proper. There are some very deep mines, but you can map the whole area and get out of any situation. Here lies the beauty of this technique: by using various functions of the type described, you can invent your own structures and explore them without knowing in advance what you will find!

It will be interesting to see what sort of programs you budding programmers come up with, now that I have laid down the foundations for a really exciting method of data storage in adventure games. So come on, develop some programs around this technique or else the Troll will get you!

```
BIG MAZES

10  X=RND(-1):X=103:Y=97:Z=1
20  T=.3+RND(8)*.25:F=0
30  IFF=1THENPRINT"Congrats you got the g
old.....!!!":END
40  GOSUB150
50  PRINT"Command  ";
60  CM$=INKEY$:IFCM$=""THEN60
70  PRINTCM$
80  IFCM$="N"THEN310
90  IFCM$="E"THEN340
100  IFCM$="W"THEN370
110  IFCM$="S"THEN400
120  IFCM$="U"THEN430
130  IFCM$="D"THEN470
140  SOUND1,110,15:FORI=0TO100:NEXT:SOUND
0:GOTO30
150  V1=INT(W*1000)
160  IFV1>35ANDV1<39ANDZ>1THENPRINT"You s
ee a crock of gold!!!":F=1
170  IFINT(RND(8)*30)=9THENPRINT"An earth
quake has just hit the area!!":T=T+.2*RN
D(8)-.1
180  IFZ=1THEN280
190  PRINT"Corridors:";:NC=0
```

```
200  X=X+1:GOSUB490: IFW<TTHENPRINT"E";:NC
=NC+1
210  X=X-2:GOSUB490: X=X+1: IFW<TTHENPRINT"
W";:NC=NC+1
220  Y=Y+1:GOSUB490: IFW<TTHENPRINT"N";:NC
=NC+1
230  Y=Y-2:GOSUB490: Y=Y+1: IFW<TTHENPRINT"
S";:NC=NC+1
240  Z=Z+1:GOSUB490: IFW<TTHENPRINT"D";:NC
=NC+1
250  Z=Z-2:GOSUB490: Z=Z+1: IFW<TTHENPRINT"
U";:NC=NC+1
260  IFNC=0THENPRINT"NONE...The earthquak
e has trapped you!"
270  PRINT:RETURN
280  PRINT"Ground level": Z=Z+1:GOSUB490: Z
=Z-1: IFFTHENF=2
290  IFW<TTHENFRINT"Shaft descends from h
ere."
300  RETURN
310  Y=Y+1: IFZ=1THEN30
320  GOSUB490: IFW<TTHEN30
330  Y=Y-1:GOTO140
340  X=X+1: IFZ=1THEN30
350  GOSUB490: IFW<TTHEN30
360  X=X-1:GOTO140
370  X=X-1: IFZ=1THEN30
380  GOSUB490: IFW<TTHEN30
390  X=X+1:GOTO140
400  Y=Y-1: IFZ=1THEN30
410  GOSUB490: IFW<TTHEN30
420  Y=Y+1:GOTO140
430  IFZ=1THEN140
440  Z=Z-1: IFZ=1THEN30
450  GOSUB490: IFW<TTHEN30
460  Z=Z+1:GOTO140
470  Z=Z+1:GOSUB490: IFW<TTHEN30
480  Z=Z-1:GOTO140
490  U=100*SQR(X*X+Y*Y*Z):W=U-INT(U):RETU
RN
```

A Mystery Program

Rod Cuckow

The following is a little mystery program, I won't say what it does. The only thing I'll hint at is that everyone will say that it is a little late!

```
10 SCREEN 2,2:CLS
20 CIRCLE (100,75),25,8,1,.48,0
30 CIRCLE (150,75),25,8,1,.49,.04
40 CIRCLE (250,0),192,8,1,.365,.428
50 CIRCLE (0,0),192,8,1,.075,.135
60 LINE (50,49)-(125,89),1
70 LINE -(125,91),1
80 LINE -(50,51),1
90 LINE -(50,49),1
100 LINE (50,50)-(125,90),1
110 LINE (160,110)-(190,127),1
120 LINE -(185,121),1
130 LINE -(183,127),1
140 LINE -(190,127),1
150 LINE (65,56)-(50,40),1
160 LINE -(50,50),1
170 LINE (65,80)-(46,56),1
180 LINE -(50,50),1
190 PAINT (125,140),8
200 GOTO 200
```

```
1170 NEXT r1
1180 IF x>y THEN 1210
1190 PRINT j$;" wins round";r
1200 j=j+1
1210 PRINT 1$;" wins round";r
1220 I=I+1
1230 PRINT:PRINT"------------------------":PRINT
1240 NEXT r
1250 IF j>=2 THEN 1310
1260 IF 1>=2 THEN 1330
1270 PRINT j$;" is out cold....":PRINT"So ";1$;" is champ!"
1280 GOTO 1340
1290 PRINT 1$;" is K.O.'d, making ";j$:PRINT"The champ"
1300 GOTO 1340
1310 PRINT j$;" wins (nice going ";j$;")."
1320 GOTO 1340
1330 PRINT 1$;" amazingly pummles ";j$;" to bits"
1340 PRINT:PRINT:PRINT:PRINT
1350 PRINT"Goodbye from Caesars Palace, Las Vegas"
```

A look at the ADVENTURE game

. . . Michael Howard

Of all the computer games that are availbale on the Sega, and many other computers for that matter, none seems to have gained the popularity that Adventure games have, save for "Space Invaders"!

What is "ADVENTURE"?

Well, let me start at the very beginning. Many years ago, a man called Gary Gygax invented a game called "Dungeons and Dragons", and many people (me included!) got caught up in the atmosphere of role-playing.

Dungeons and Dragons was the first role playing game in which players assume alter egos and use them to wander through strange, exotic lands or caves and encounter many monsters and strive to accumulate wealth while building up his/her character. Dungeons and Dragons has it's origins in JRR Tolkien novels, such as "The Hobbit", "The Lord of the Rings" and "The Silmarillion".

These games usually have a "Dungeon Master" who does not actually participate in the game. He or she has a detailed map of the dungeon, and charts the players progress (and other nasty creatures) through it. The players tell the Dungeon Master where they wish to go, and the Dungeon Master describes their new location in great detail, along with objects they may see. If a confliction with a creature takes place, the Dungeon Master rolls dice to simulate the random element in any conflict, and after weighing up such factors as what weapons are being used, how strong the fighters are etc — the result of the conflict is announced. As you can see there are few limitations to the game and some games last for ages. (One game, of which I am the Dungeon Master, has lasted over 2 YEARS!!!).

When computers came on the market and became popular, a gentleman by the name of Will Crowther decided to write a game that simulated the Dungeons and Dragons situation, with the computer being the Dungeon Master. The program was written in a now archaic language called "Fortran" and was called, quite simply, Adventure. The program was released in 1976 and was added to the "DECUS" library, (the DECUS library is for DEC PDP-11 (a large computer!) users) and library. The game was so popular that people were phoning up the programmer at 3am to ask how to get over the Troll's bridge.

Unfortunately, in those days, games such as Adventure needed extensive disc backup, owing to the lack of on-board memory. Well, now computers have quite large memorys and are designed perfectly for adventure games.

Luckily, we now have quite a few adventure games available for the Sega, thanks to the likes of Flexisoft, Scorpion Software and Poseidon Software, and we have a few left. So hopefully, this little section will spark some interest in such games.

The games to look at are The Youngins, TTS, Transylvania Castle of horror, Underworld of Kragon, Castaway, The House! And others.

So get stuck in!

Acey Ducey

This program is a simulation of the card game Acey Ducey. In the game, the dealer (in this case the computer) deals two cards face up. You have an option to have a value between the first two.

Your initial money is set to $100. The game continues until you are skint or you switch the power off!

Here goes . . .

```
10 CLS:PRINT "ACEY DUCEY":X=RND(-1)
20 PRINT
30 PRINT "ACEY DUCEY IS PLAYED IN THE FOLLOWING MANNER "
40 PRINT "THE DEALER DEALS TWO CARDS FACE UP"
50 PRINT "YOU HAVE THE OPTION OF BETTING OR NOT BETTING"
60 PRINT "DEPENDING ON WHETHER YOU FEEL THE NEXT CARD"
70 PRINT "WILL HAVE A VALUE IN BETWEEN THE FIRST TWO"
80 PRINT "IF YOU DO NOT WANT TO BET THEN INPUT 0"
90 N=100
100 Q=100
110 PRINT "YOU HAVE $";Q
120 PRINT
130 GOTO 180
140 Q=Q+M
150 GOTO 110
160 Q=Q-M
170 GOTO 110
180 PRINT "HERE ARE YOUR NEXT TWO CARDS.."
190 A=INT(14*RND(1)+2)
200 IF A<2 THEN 190
210 IF A>14 THEN 190
220 B=INT(14*RND(1)+2)
230 IF B<2 THEN 220
240 IF B>14 THEN 220
250 IF A>=B THEN 190
260 IF A<11 THEN 310
270 IF A=11 THEN 330
280 IF A=12 THEN 350
290 IF A=13 THEN 370
300 IF A=14 THEN 390
310 PRINT A
320 GOTO 400
330 PRINT "JACK"
340 GOTO 400
350 PRINT "QUEEN"
360 GOTO 400
370 PRINT "KING"
380 GOTO 400
390 PRINT "ACE"
400 IF B<11 THEN 450
410 IF B=11 THEN 470
420 IF B=12 THEN 490
430 IF B=13 THEN 510
440 IF B=14 THEN 530
450 PRINT B
460 GOTO 550
470 PRINT "JACK"
480 GOTO 550
490 PRINT "QUEEN"
500 GOTO 550
510 PRINT "KING"
520 GOTO 550
530 PRINT "ACE"
540 PRINT
550 PRINT
560 INPUT "WHAT IS YOUR BET";M
570 IF M<>0 THEN 600
580 A$="CHICKEN":IF RND(8)>0.5 THEN A$="TURKEY!"
590 PRINT A$:GOTO 180
600 IF M<=Q THEN PRINT:PRINT"NEXT CARD..":GOTO 640
```

```
610 PRINT "YOU'LL BE LUCKY, YOU DON'T HAVE THAT MUCH!"
620 PRINT "YOU HAVE A MERE $";Q
630 GOTO 550
640 C=INT(14*RND(1)+2)
650 IF C<2 THEN 640
660 IF C>14 THEN 640
670 IF C<11 THEN 720
680 IF C=11 THEN 740
690 IF C=12 THEN 760
700 IF C=13 THEN 780
710 IF C=14 THEN 800
720 PRINT C
730 GOTO 820
740 PRINT "JACK"
750 GOTO 820
760 PRINT "QUEEN"
770 GOTO 820
780 PRINT "KING"
790 GOTO 820
800 PRINT "ACE"
810 PRINT
820 IF C>A THEN 840
830 GOTO 870
840 IF C>=B THEN 870
850 PRINT "YOU WIN!!!!!!!"
860 GOTO 140
870 PRINT "HA HA SUCKER...YOU LOSE!"
880 IF M<Q THEN 160
890 PRINT
900 PRINT
910 PRINT "SORRY, YOU'VE NO MONEY LEFT!!"
```

Crazy Cancellation . . . More maths fun!!

$$\frac{x^2 + 7x + 12}{x^2 + 4x + 3} = \frac{(x+4)(x+3)}{(x+1)(x+3)} = \frac{x+4}{x+1}$$

Last months little maths program showed that maths is great fun on computers (well sometimes!), and suprisingly, quite a few people wrote to me asking for more maths fun. So I decided to oblige!

The following program finds incidences of coincidentally correct cancellations! What-a-mouthful! Basically, all that means is . . . well let me show you an example:-

16/64 = 1/4, by cancelling the sixes . . . see what I mean!?

Not only is the maths correct, but the fact that some common numbers can be cancelled and still produce the same result . . . is quite amazzzzzzing!

Oh by the way, fractions such as 10/20 and 20/30 are ignored because they are obvious and thus very, very boring!

All the fractions are less than 1. One of the main drawbacks of the program is that it is very slow.

Here is the program:

```
1 REM MH
10 er=0.0001
20 FOR t=1 TO 9
30 FOR u=1 TO 9
40 FOR a=1 TO 9
50 FOR b=1 TO 9
60 IF t<>a AND t<>b AND u<>b THEN 240
70 nu=10*a+b
80 de=10*t+u
90 IF nu>de THEN 130
100 b=9
110 a=9
120 GOTO 240
130 q=nu/de
140 e1=ABS(qpa/t)
150 e2=ABS(q-a/u)
160 e3=ABS(q-b/t)
170 e4=ABS(q-b/u)
180 IF a=t AND e4<er THEN 230
190 IF a=u AND e3<er THEN 230
200 IF b=t AND e2<er THEN 230
210 IF b=u NAD e1<er THEN 230
220 GOTO 240
230 PRINT nu;"/";de; "is a solution."
240 NEXT b
250 NEXT a
260 NEXT n
270 NEXT t
```

Olympic Boxing

This program simulates a boxing match of three rounds. The computer coaches one of the boxers and determines his defences and best punches, while you do the same for your boxer. At the start of the match, you must specify your boxer's best punch and his vulnerability.

There are approximately seven major punches per round, although this may be varied by line number 250. The best of 3 rounds wins the championship.

```
10 SCREEN 1,1:CLS
20 PRINT"Boxing... 2 out of 3 wins"
30 PRINT:PRINT:PRINT
40 x=RND(-1)
50 j=0:l=0
60 INPUT"What is your opponent's name ";j$
70 INPUT"What is your boxer's name ";l$
80 PRINT"Different punches are "
90 PRINT"1..Full swing"
100 PRINT"2..Hook to the jaw"
110 PRINT"3..Uppercut"
120 PRINT"4..Jab"
130 PRINT"What is your boxer's best punch";:INPUT b
140 PRINT"What is his vulnerability";:INPUT d
150 b1=INT(4*RND(8)+1)
160 d1=INT(4*RND(8)+1)
170 IF b1=d1 THEN 150
180 PRINT j$;"'s advantage is";b1:PRINT"And his vulnerability is a secret."
190 FOR r=1 TO 3
200 IF j>=2 THEN 1310
210 IF l>=2 THEN 1330
220 x=0:y=0
230 PRINT"Round";r;" begins..."
240 PRINT
250 FOR r1=1 TO 7
260 i=INT(10*RND(8)+1)
270 IF i>5 THEN 760
280 PRINT l$;"'s punch";
290 INPUT p
300 IF p=b THEN 320
310 GOTO 330
320 x=x+2
330 IF p=1 THEN 440
340 IF p=2 THEN 580
350 IF p=3 THEN 670
360 PRINT l$;" jabs at ";j$;"'s head ";
370 IF d1=4 THEN 400
380 c=INT(8*RND(8))+1
390 IF c<4 THEN 420
400 x=x+3
410 GOTO 1170
420 PRINT"It's blocked."
430 GOTO 1170
440 PRINT l$;" swings and ";
450 IF d1=4 THEN 540
460 x3=INT(RND(8)*30)+1
470 IF x3<10 THEN 540
480 PRINT"He misses ";
490 PRINT
500 IF x=1 THEN 1170
510 PRINT
520 PRINT
530 GOTO 410
```

```
540 PRINT"He connects!!!"
550 IF x>35 THEN 1270
560 x=x+15
570 GOTO 410
580 PRINT l$;" tries a hook....";
590 IF d1=2 THEN 630
600 h1=INT(2*RND(8)+1)
610 IF h1=1 THEN 650
620 PRINT"connects.."
630 y=y+7
640 GOTO 410
650 PRINT"But it's BLOCKED!!!!!!!"
660 GOTO 410
670 PRINT l$;" goes for an uppercut  ";
680 IF d1=3 THEN 730
690 d5=INT(RND(8)*100)+1
700 IF d5<51 THEN 730
710 PRINT" and it's blocked (lucky block!)"
720 GOTO 410
730 PRINT"and he connects!"
740 x=x+4
750 GOTO 410
760 j7=INT(4*RND(8)+1)
770 IF j7=b1 THEN 790
780 GOTO 800
790 y=y+2
800 ON j7 GOTO 900,1000,1070
810 PRINT j$;" jabs and ";
820 IF d=4 THEN 880
830 z4=INT(7*RND(8)+1)
840 IF z4>4 THEN 870
850 PRINT" It's blocked!"
860 GOTO 410
870 PRINT" blood spills!!"
880 y=y+5
890 GOTO 410
900 PRINT j$;" takes a full swing and ";
910 IF d=1 THEN 960
920 r6=INT(60*RND(8)+1)
930 IF r6<30 THEN 960
940 PRINT" But it's blocked !"
950 GOTO 410
960 PRINT" POW!! He hits him right in the face!"
970 IF y>35 THEN 1290
980 y=y+15
990 GOTO 410
1000 PRINT j$;" gets ";l$;" in the kisser!"
1010 y=y+7
1020 PRINT"... and again!"
1030 y=y+5
1040 IF y>35 THEN 1290
1050 PRINT
1060 GOTO 1240
1070 PRINT l$;" is attacked by an uppercut.."
1080 IF d=3 THEN 1110
1090 q4=INT(200*RND(8)+1)
1100 IF q4>75 THEN 1140
1110 PRINT" and ";j$;" connects.."
1120 y=y+8
1130 GOTO 410
1140 PRINT" blocks and hits ";j$;" with a hook."
1150 x=x+5
1160 GOTO 410
```

Bowls

This is a simulated bowling game for up to four players. You play ten frames.

To roll a ball, you simply type 'roll'. After each roll, the Sega will show you a diagram of the remaining pins, a 'O' represents a pin still standing, and the computer will give you the folliwng analysis:

Gutter . . . the ball went down the gutter!
Strike . . . congratulations!
Spare . . . you made a boo boo!
Error . . . this only happens if you made two rolls and missed on both ocassions.

```
1 REM By Mark Goodwin 1986
10 CLS:PRINT"...Bowling.."
20 PRINT:PRINT:PRINT
30 DIM c(15),a(100,6)
40 PRINT"Welcome to the alley"
50 PRINT"Bring your mates"
60 PRINT:PRINT"Okay let's first get acquainted"
70 PRINT:PRINT
80 PRINT"First of all, how many are playing"
90 INPUT r
100 PRINT
110 PRINT"Very good...!"
120 FOR j=1 TO 6:FOR i=1 TO 100:a(i,j)=0:NEXT i,j
130 f=1
140 FOR p=1 TO r
150 m=0
160 b=1
170 m=0:q=0
180 FOR i=1 TO 15:c(i)=0:NEXT i
190 REM
200 PRINT"Type 'ROLL' to get the ball going."
210 INPUT n$
220 k=0:d=0
230 FOR i=1 TO 20
240 x=INT(RND(8)*100)
250 FOR j=1 TO 10
260 IF x<15*j THEN 280
270 NEXT j
280 c(15*j-x)=1
290 NEXT i
300 REM
310 PRINT"Player:";p;" Frame:";f;" Ball:";b
320 FOR i=0 TO 3
330 PRINT:PRINT TAB(i+1)
340 FOR j=1 TO 4-i
350 k=k+1
360 IF c(k)=1 THEN 390
370 PRINT "O ";
380 GOTO 400
390 PRINT "  ";
400 NEXT j
410 NEXT i
420 PRINT
430 REM
440 FOR i=1 TO 10
450 d=d+c(i)
460 NEXT i
470 IF d-m<>0 THEN 490
480 PRINT"In the Gutter!"
490 IF b<>1 OR d<>10 THEN 520
500 PRINT"!!* STRIKE *!!"
```

```
510 q=3
520 IF b<>2 OR d<>10 THEN 550
530 PRINT"SPARE!!"
540 q=2
550 IF b<>2 OR d>=10 THEN 580
560 PRINT"ERROR!"
570 q=1
580 IF b<>1 OR d>=10 THEN 600
590 PRINT"Roll your 2nd ball."
600 REM
610 PRINT
620 a(f*p,b)=d
630 IF b=2 THEN 690
640 b=2
650 m=d
660 IF q=3 THEN 600
670 a(f*p,b)=d-m
680 IF q=0 THEN 190
690 a(f*p,3)=q
700 NEXT p
710 f=f+1
720 IF f<11 THEN 140
730 PRINT"Frames"
740 FOR i=1 TO 10
750 PRINT i;
760 NEXT i
770 PRINT
780 FOR p=1 TO r
790 FOR i=1 TO 3
800 FOR j=1 TO 10
810 PRINT a(j*p,i);
820 NEXT j
830 PRINT
840 NEXT i
850 PRINT
860 NEXT p
870 PRINT"Do you want another game."
880 INPUT a$
890 IF LEFT$(a$,1)="Y" OR LEFT$(a$,1)="y" THEN 10
```

The Bouncing Ball . . . a study of Kinematics

This program tests your fundamental knowledge of kinematics. It presents a simple problem: a ball is thrown straight up in the air at some random speed (velocity). You then must answer three questions about the flight of the ball:

1. How high will it go?
2. How long until it returns to Earth?
3. What will be it's velocity after a random number of seconds?

The computer evaluates your performance; within 15% of the correct answer is considered close enough. After each run, the computer gives you another problem until you RESET or BREAK the program.

```
10 CLS
20 PRINT"Kinematics"
30 PRINT:PRINT:PRINT
40 PRINT
50 q=0
60 v=5+INT(35*RND(8))
70 PRINT"The ball is thrown upwards at";v;" m/s"
80 PRINT
90 a=0.05*v^2
100 PRINT"How high will it go (in meters)";
110 GOSUB 240
120 a=v/5
130 PRINT"How long until it returns (in secs.)";
140 GOSUB 240
150 t=1+INT(2*v*RND(8))/10
160 a=v-10*t
170 PRINT"What will its velocity be after ";t:PRINT"Seconds."
180 GOSUB 240
190 PRINT
200 PRINT q;" right out of 3.";
210 IF q<2 THEN 40
220 PRINT"  not bad."
230 GOTO 40
240 INPUT g
250 IF ABS((g-a)/a)<0.15 THEN 280
260 PRINT"Not even close...."
270 GOTO 300
280 PRINT"Close enough."
290 q=q+1
300 PRINT"Correct answer is";a
310 PRINT:RETURN
```